Experiences From Constructing Command and Control Simulations Using a Tactical Data Link Standard

Dirk C. Uys and Arno Duvenhage
Council for Scientific and Industrial Research
Meiring Naude Road
Pretoria, 0001
South Africa
+27 12 841 4758/3319
duys@csir.co.za, aduvenhage@csir.co.za

August 13, 2009

ABSTRACT: *Most operational Command and Control systems use a Tactical Data Link to communicate. The need to integrate simulated systems with operational systems and other simulated systems resulted in the utilization of Tactical Data Link for the integration. Integrating simulated systems in this way are not necessarily ideal and presents several challenges with regards to data distribution and time management. Data model translation and the exchange of simulation meta-data also present some difficulties. The observations made and problems encountered while integrating different simulators using a tactical data link are discussed in this paper.*

1 Introduction

Command and Control (C2) simulations can be used to evaluate existing or new C2 processes. Typically a C2 simulation will contain some simulated systems as well as some real world software and hardware systems (referred to as operational systems in the rest of this paper). These different systems must be able to communicate with each other. It is necessary to configure C2 simulations to be capable of running with various combinations of operational and simulated systems.

There are several technologies specifically aimed at simulation interoperability (such as the High Level Architecture (HLA) [1]), but unfortunately it is not always feasible to add functionality in existing systems to support these technologies.

A practical solution for integrating operational and simulated systems is to use a Tactical Data Link (TDL) standard since many operational systems support some sort of TDL to communicate with other systems. More often than not the TDL have the ability to connect over a local area network (LAN) using TCP/IP, even though the TDL standard might be designed to use another communication medium.

Using a TDL standard to integrate simulated and

operational systems is not necessarily ideal and pose a few challenges. These include the management of time, translation between different data models and data distribution.

Since TDLs are designed for the transportation of C2 information within a specific environment [5], the operational doctrine that applies to that environment influences the TDL message set and structure. This places limits what information can be exchanged and how information can be transported over the TDL. This may have severe implications on new C2 processes being evaluated.

The paper proceeds by describing the work done by The Council for Scientific and Industrial Research (CSIR) on integrating operational and simulated systems. The challenges when using a TDL standard for this are discussed and explained with the lessons learned from the integrating of systems. This leads to some suggestions on things to consider when using a TDL standard for simulation interoperability.

2 Background

The CSIR have, during the past few years, been building a C2 simulation and interoperability capability aimed at supporting joint operations within the South African National Defense Force. During this time the capability has been deployed at various field exercises to either exercise the command structures with simulated systems or to help integrate the C2 systems and equipment involved in the exercise.

TDLs in general enable the creation of a network of nodes that can exchange information with each other. This information is normally encoded with a fixed set of text or binary messages. The messages are normally exchanged over radio frequencies, but some TDLs also support other transport mediums like local area networks (LANs). There are several well established TDLs like the NATO Link11 and Link16. The TDL standard used by the CSIR for integration of systems was specifically developed for the South African National Defense Force and is similar to Link11.

The TDL standard implementation required to inter-operate with the C2 systems during field exercises had to be incorporated into the CSIR's simulation software. This part of the software is called the *C2 Protocol Gateway* and discussed in [**?**]. This allowed the software to interface with the relevant C2 systems and exchange data with them using the TDL standard. This worked well as long as the simulation software could manage real-time execution and didn't fall behind.

Other software systems and simulators from industry also implemented TDL standards in order to interface with C2 systems. The TDL implementations for most of these simulators operate on Local Area Networks (LANs) using TCP/IP or UDP to connect. Since the simulators all had compatible interfaces, the TDL standard could be used to connect different virtual or constructive simulators together. If no live systems are present the simulations are free to run either faster or slower than real-time or even pause. Unfortunately the TDL standard does not accommodate this *freedom*. These and other challenges are discussed in the following section.

3 Challenges

Using a TDL standard to integrate operational and simulated systems presents several challenges. These challenges involve time management, data model translation and the distribution of data. In

this section these challenges are discussed by giving a generic overview of the problem, a practical example of how it affected a real experiment and then possible ways of overcoming the problem (if any).

3.1 Time Management

Simulation interoperability technologies like HLA [1] have formal ways to manage the progression of time during a simulation [3]. Most TDL standards however do not explicitly support time management. With systems running at the same speed and instantiation passing of messages between systems, there is little need for time management more comprehensive that syncing the system clocks. But when messages can possibly be delayed and certain systems may be held up for a while, time management becomes necessary.

Figure 1 illustrates a possible problem that may arise from delayed messages. Three different systems (S_1, S_2 and S_3) are running concurrently. At time t_0 S_1 and S_3 both send a message, M_1 and M_2 respectively to S_2. For some reason M_2 are delayed one time-step longer than M_1. By the time S_2 receives both messages, it already advanced two time steps. Timestamps can help system S_2 to determine that this happened, but there is no guarantee that other messages sent at time t_0 aren't also delayed and still to be delivered. It is thus not trivial for S_2 to know when it may advance to the next time step.

When messages sent at the same time step are received at different time steps, it is possible that the original order of the messages messages can be corrupted. A message can arrive at a system before other messages that was sent before it. Some systems may be sensitive to the order messages are received in as it can influence causal effects. The

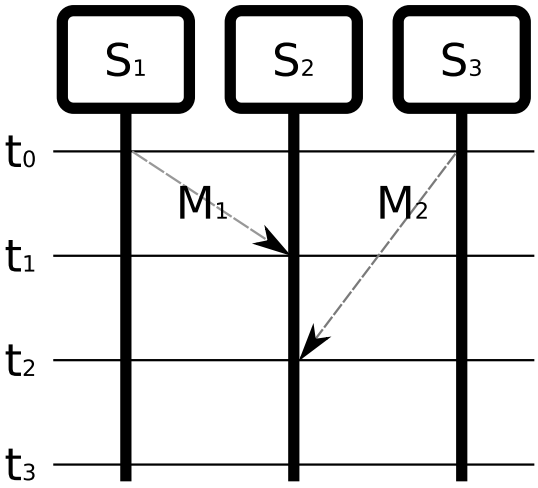

Figure 1: Time management

integrity of the simulations results may be affected by this breakdown in causality.

If one or more of the systems start to run slower than real-time it can also cause problems. Other systems may be unaware of this and continue running at normal speed while the other system starts lagging further behind.

Running a simulation faster than real-time does not always make sense, but sometimes it is desirable to run the simulation, or parts thereof, faster than real-time. Again most TDLs do not have the capability to inform other systems when to run faster than real-time and there is no inherent support to synchronize the systems.

In one of the field exercises, some systems started running slower than real-time while the other systems continued running at real-time. After a while, one of the systems lagged very far behind. This system was responsible for producing simulated radar tracks. The tracking filters on some of the other systems interpreted the lack of recent track data to indicate that the track ceased to exist. Tracks that were of importance to the exercises simply disap-

peared. Fixing such a problem is not trivial.

One way of overcoming this problem is to establish a secondary link with other systems [6]. This link can then be used to regulate the passing of time in the different systems. Unfortunately it is not always possible to develop this additional functionality for all the involved systems. Using a secondary link that needs to be implemented by all other systems negate some of the benefits gained when using a TDL for integration.

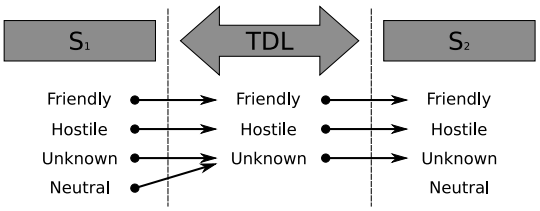

Figure 2: Data mapping

3.2 Data Model Translation

A TDL supports a message set developed for a specific C2 context. This message set implicitly defines the data model used for data exchange. The TDL data model might not map perfectly onto the data models used by different systems. The information passed between the systems have to be mapped onto the message set and the implied data model of the TDL.

There could be some loss of information because of an imperfect mapping. This has the effect that information sent over the TDL from one system to another and back again could arrive slightly different. Even if the structure of the specific TDL messages used for interoperability matches that of the used data model, the representation of attributes in the messages can still be interpreted differently. This can include spatial references, distance measures and classification values.

Figure 2 shows an example where a system's data model has more affiliation values than the corresponding TDL standard attributes have: *neutral* is understood as *unknown* by the TDL and translated back to *unknown* and not *neutral* in the second system.

This could be seen from a case where an aircraft that was classified as a neutral military fixed wing in one system, showed up in another instance of the same system as a civilian aircraft. In this case, it presented as a small discrepancy, but a error like this can cause integration errors that are hard to solve.

Because TDLs are developed to exchange C2 information, the message set is normally tailored to the current doctrine of the C2 environment. Testing new doctrine using a TDL that already incorporates current doctrine in its message set is problematic as the doctrine specifies what messages there should be and what information they should contain. In order to satisfy the requirements for an experiment it may be necessary to send more than one message or to send a message that contains irrelevant information in order to communicate the desired data.

It may be possible to communicate information not captured by the message set of the TDL by using a free text messages. Information can be encoded into text and then embedded in the free text message. This implies that the system receiving the message should be aware of this and scan free text messages for encoded information. Both sending and receiving system need to be aware of this type of exchange.

3.3 Data Distribution

Not all TDLs support one-to-one, one-to-many and many-to-many communication [2]. This does not prevent integration, but it might complicate the transfer of data and the management of connections. Figure 3 shows one node (N_1) broadcasting a message (M_1) to all the other nodes on the network. If the links between the nodes are one-to-one links, the message needs to be broadcast four times over different links. Traditionally this is not a major concern for TDLs but in a simulated environment it is not always desirable for a system to know what other systems it must send its data to.

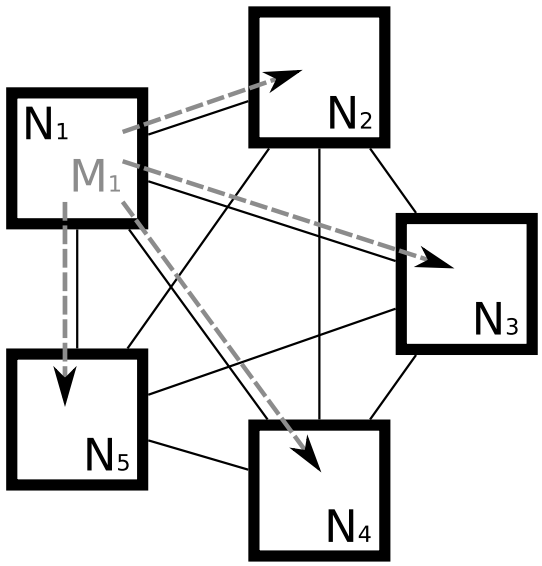

Figure 3: Data Distribution

TDLs are meant for communicating the latest updates and do not keep historical data [5]. In most situations this is sufficient, but some systems require initial state or historical data to be obtained when they connect for the first time. TDL messages could be used to request the initial state or historical data, but they may the be used differently from how the standard stipulates. This makes the implementation incompatible with the TDL implementations of operational equipment and systems that use the standard correctly.

Some of the exercises required an instant messaging service to be implemented. It should be able to send messages to a single destination or to broadcast the message to all interested parties. Setting this up using a TDL presented some difficulty. The TDL standard that was used only supported one-to-one or broadcast communication and not many-to-many communication where a message could be sent to a group. A separate link had to be set up between every entity to obtain the required result. This solution proved to be tedious when there was a large number of entities.

3.4 Standard Compliance

As with any standard in the real world, not all systems supporting the TDL standard implemented the same version of the standard. Other systems implemented the standard incorrectly.

Some of the messages described by the standard required an acknowledgment to be sent upon reception. Not all systems implemented this acknowledgment while other systems did. The result was that one system kept waiting for the acknowledgment from another system. Since the other system did not implement the response, the response was never sent and the first system kept on waiting without responding to other messages from other systems. Further communication with the first system was not possible anymore.

4 Conclusion

TDLs are already used to integrate different operational systems. It is also possible to use these TDLs

to integrate simulated systems with other simulated systems or with operational systems. This can provide a practical solution that can be applied with modest effort.

It is however important to know that such a solution imposes certain limitations and introduces some problems. Awareness of these limitations and problems are very import when trying to ensure that simulations produce accurate and reliable results. Failure to address them may be lead to incorrect conclusions being drawn from experiments.

Because the message set supported by a TDL normally corresponds to the tactical doctrine that applies to the C2 environment where the TDL is used, it may prevent the evaluation of concepts outside the existing doctrine. Many times this is the very purpose of constructing C2 simulations. If it is not possible to use the message set for a given experiment, some other means for integration might prove to be more fruitful.

5 Future Work

Future research may be conducted to determine possible changes that can be made to a TDL standard to address the challenges discussed in this paper. This could strengthen the case for using TDL standards for simulation interoperability. Almost in line with this, future research may also be conducted on possible amendments to the TDL standard for better operation over TPC/IP networks.

Acknowledgments

The authors would like to thank both the Armaments Corporation (Armscor) of South-Africa and the Council for Scientific and Industrial Research (CSIR) of South-Africa for supporting this research.

References

[1] *IEEE1516, High Level Architecture (HLA)*, Mar. 2001. www.ieee.org.

[2] A. Filippidis, T. Doan, and B. Tobin. Net warrior - DTSO Battlelab interoperability. In *Proceedings of the 2007 SimTecT Conference*, pages 93–98, June 2007.

[3] R. M. Fujimoto. Time Management in The High Level Architecture. *SIMULATION*, 71(6):388–400, 1998.

[4] F. Hill. Systemic problems with data link simulation. In *Proceedings of the 2003 Fall Simulation Interoperability Workshop (FALL-SIW 2003)*, 2003.

[5] P. G. Larsen. Coalition C2 interoperability challenges. In *Proceedings of the 11th International Command and Control Research and Technology Symposium*, 2006.

[6] J. J. Nel, W. H. le Roux, O. van der Schyf, and M. Mostert. Modelling joint air defence doctrinal issues with a linkza-based integration of two C2 simulators: A case study. In *In Proceedings of the 2007 Military Information and Communications Symposium of South Africa (MICSSA)*, Pretoria, July 2007. Presented by Nel (Paper Number 1-03B-2).

DIRK UYS is a Researcher for the Council for Scientific and Industrial Research (CSIR), South Africa. He joined the CSIR's Mathematical and Computational Modelling Research Group in 2008 as a software developer. Dirk has a Honours Degree in Computer Science from the University

of Pretoria, South Africa.

ARNO DUVENHAGE is a Researcher for the Council for Scientific and Industrial Research (CSIR), South Africa. He joined the CSIR's Mathematical and Computational Modelling Research Group in 2005 as a Software Engineer. Arno's current work involves modelling and simulation for decision support, focusing on joint operations, specializing in distributed and networked systems. Arno has a BEng Degree in Computer Engineering from the University of Pretoria, South Africa, and is currently busy with a Masters in software engineering.