Configuration and Initialization – Process Definition and Tool Development Within Euclid RTP 11.13 (03E-SIW-059)

Jan Jelle Boomgaardt
TNO Physics and Electronics Laboratory.
P.O. Box 96864
2509 JG The Hague, The Netherlands
Email: boomgaardt@fel.tno.nl

Magnus Johansson
Pitch Kunskapsutveckling AB
Nygatan 35, 58219 Linköping, Sweden
Email: magnus.johansson@pitch.se

Keywords:
Euclid11.13, Simulator Architecture, HLA, Tool Support, Configuration and Initialization

ABSTRACT: *Euclid RTP 11.13 is a major European initiative to promote the use of Synthetic Environments (SEs). The aim of the project is to "overcome the obstacles that prevent SEs being exploited in Europe by developing a process and an integrated set of prototype tools that will reduce the cost and time-scale of specifying, creating and utilizing SEs for collective training, mission rehearsal and simulation based acquisition". The SE Development & Exploitation Process (SEDEP) is the EUCLID RTP 11.13 process that has been developed for creating and utilizing SEs. SEDEP is based on FEDEP, but extends it with respect to improved support for iterative federation development and re-use of existing components, which are stored in a Repository. The Repository lies at the heart of the SEDEP and provides the facility for storing and sharing data about a SE project.*
Federate Configuration and Initialization (FC&I) is a vital part when preparing a federation. Within the SEDEP, Configuration is defined as the process of creating instances of scenario objects (e.g. computer-generated forces), while the Initialization process assigns an initial value or start condition to those scenario objects.
Configuration and Initialization lies in the SEDEP step "Integrate & Test Federation" to support an activity called "Prepare Federation". The SEDEP process describes that the purpose of this step is "to initialize and configure the identified federates, establish all required interconnectivity between federates, and test federates and the federation prior to execution".

A supporting tool (FC&I tool) has been developed to automate the preparation of a federation. A major design decision for the FC&I tool was that it should support the concept of hierarchical nested federations. The design and implementation of the FC&I tool fully supports the concept of component-based federates. A key element of the tool's operation is the HLA ownership transfer mechanism.
This paper describes our views regarding Configuration and Initialization to prepare a federation, a (component-based) federate and its components. The paper also discusses the design and implementation of the tool and the developed test environment.

1. Introduction

Federate Configuration and Initialization (FC&I) is a vital part when preparing a federation. Configuration is defined as the process of creating instances of scenario objects (e.g. computer-generated forces), while the Initialization process assigns an initial value or start condition to those scenario objects. See figure 1.

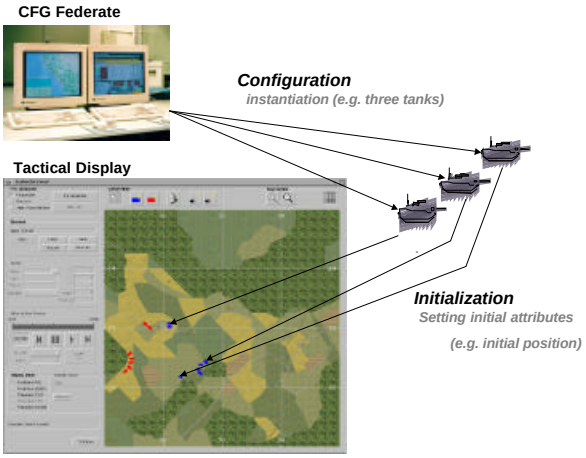

Figure 1: Preparing a Scenario

Setting up a scenario for a Synthetic Environment (SE) is often a laborious task. In order to minimize simulation run-time, offline preparation should be promoted. Preparation includes activities like setting the begin-conditions of your synthetic environment scenario by configuring what entities are to be present and initialize them to start at predefined locations with scenario specific parameters (e.g. position, ammunition supply). See a typical example in Figure 1.

The Federate Configuration and Initialization (FC&I) tool presented in this paper activates the settings, that were prepared offline, from a centralized console in an automated fashion.

The challenges of preparing a federation have been addressed by Euclid11.13. This program is a major European initiative to promote the use of Synthetic Environments (SEs). This is done by the development of two major products. The first product is the SE Development & Exploitation Process (SEDEP), a process for creating and utilising SEs. The second Euclid 11.13 product is the SE development environment.

The SEDEP is based on FEDEP, but extends it by providing a better interface with the user and enhances it with respect to improved support for iterative federation development and re-use of existing components. SEDEP consists of eight steps. See also Figure 2.

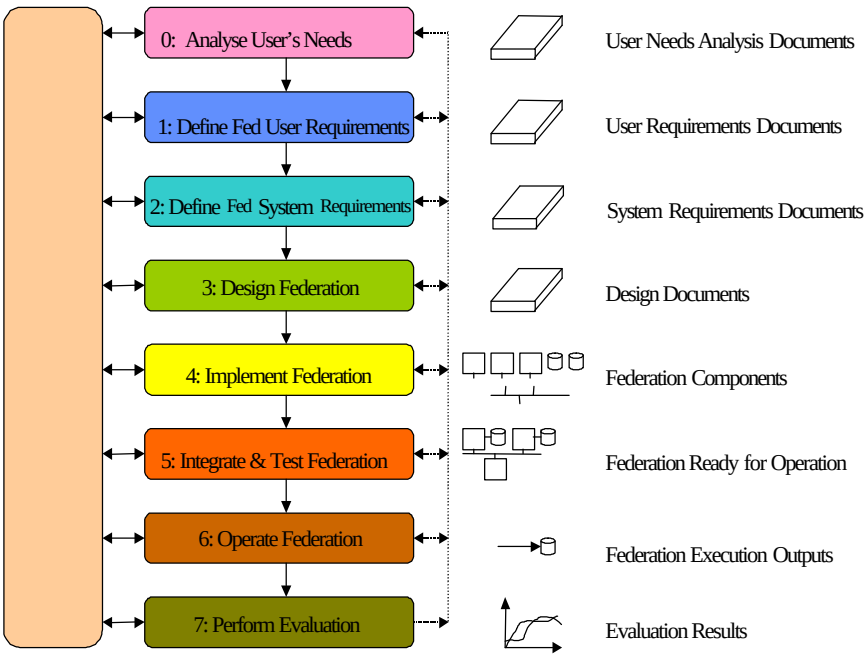

Figure 2: SEDEP Steps and Products

Where appropriate, the SEDEP can be iteratively applied to support an SE project. The depth to which the process is applied during each iteration is dependent on the quality of information provided. An important feature of the SEDEP is that all the information is stored in the Repository, so that successive iterations of the process can build on the information already available. The recent update of the IEEE FEDEP has included concepts developed within the SEDEP.

The SE development environment consists of a large set of tools, all aiding the process of SE development. At the centre of the environment lies the Repository, which ties all the tools and processes together. It is the combination of all the different components that enables the facilitation of producing and reusing future and past SEs. For more information about Euclid RTP 11.13, see [6] and [7].
The activities relating to configuration and initialization are covered in step 5 and step 6 of the SEDEP that are dealing with "Integrate & Test Federation" and Operate Federation. Step 5 is responsible for integrating the Federation elements with the RTI and testing the Federation ready for operation. Step 5 covers also an activity called "Prepare Federation". The SEDEP process describes that the purpose of this step is "to initialize and configure the identified federates, establish all required interconnectivity between federates, and test federates and the federation prior to execution". Step 6 deals with the actual operation of the simulation producing execution outputs. The FC&I tool helps to ease the execution of simulations using a synthetic environment according to the SEDEP process.

The development of the FC&I tool was a joint effort of Pitch from Sweden and TNO-FEL from the Netherlands. Pitch and TNO shared the responsibility for the analysis and design while Pitch has made a Java based implementation. The tool has been tested at TNO using a distributed multi-platform test-environment, which includes monolithic federates and component-based federates.
Any HLA compliant federate or component can benefit from this tool support if some specific interactions and services are implemented. This prerequisite is referred to as being "SEDEP-enabled".

Chapter two of this paper discusses the concept of component-based federates. Chapter three deals with preparing a federation for scenario execution. In chapter four some design considerations are discussed and design decisions are explained. The final chapters give some implementation details and conclusions.

2. Supporting Component-Based Federates

Although not explicitly defined in SEDEP, the Euclid 11.13 project encouraged support for Component-based federates. In the same way that a federation is composed of federates, a federate itself can be composed of components (i.e. sub-federates). Considering component-based federates is an Euclid 11.13 extension to the existing concepts of HLA. Supporting component-based federates is an additional challenge because component-based federates hide their internal construction (i.e. façade design pattern). However, peeking inside a component-based federate shows that configuration and initialization can benefit from the same support offered also at the higher federation level. See figure 3.

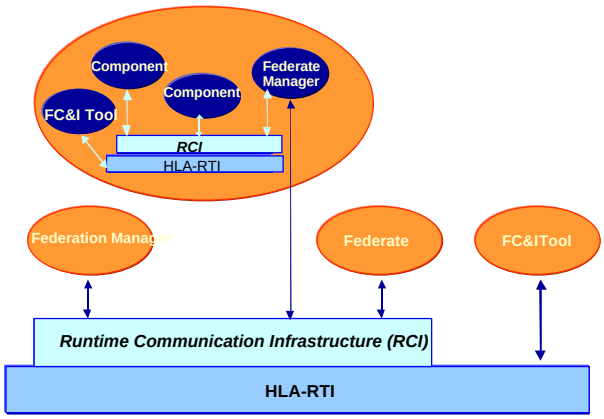

Figure 3: FC&I Component Based Test Environment

Handling component-based federates is an additional requirement that is based on ongoing work in this area. The first functional proof of this component-based concept was presented in a large demo in June 1999 at TNO-FEL in a project called SIMULTAAN. A data distribution system was introduced called the Run-time communication Infrastructure (RCI). The RCI shields components from the intra-[MJ1]federate protocol and network hardware. Another demonstration of the architecture was developed by TNO-FEL and demonstrated at ITEC2000 in The Hague, showing the Component approach for a generic fighter simulator, called the TNO Generic Strike Fighter Federate (GSF). The GSF federate was further improved and extended to demonstrate its capabilities on behalf of application for the Joint Strike Fighter (JSF) project. The results were presented as the JSF Simulation Architecture (JSA). (See [4])

Based on these previous results in the area of component-based development, an additional federate is introduced to manage the online progress of states after joining. See also figure 4. This federate is referred to as the Federation Manager (FNM). See the structure of the FC&I test-environment in [MJ2]Figure 3. The functionality of the Federation Manager could also be integrated into other federates. For example a federate that acts like an Instructor Operator System (IOS).

In case of a component-based federate, an additional component manages the interactions between the components and the parent federation. This mandatory component is referred to as a Federate Manager (FM). A Federate Manager acts like a bridge between the federation level and the component level and must join both.

3. Preparing a Federation For Scenario Execution

Developing a Federation for the execution of a scenario relates to various steps in the SEDEP process. Based on a scenario, the conceptual model and the requirement for a federation, candidate federates/components are selected (re-used) or developed. These federates/components are responsible for objects (entities) that play a role during scenario execution. We distinguish offline and online preparation. Offline preparation covers those activities that result in a list of settings specifying scenario related begin-conditions, (i.e. FC&I settings). Online preparation, on the other hand, is characterized by effectuating those FC&I settings. Offline preparation of a scenario is covered by step five: "Integrate and Test Federation" SEDEP step, while the online preparation of federates/components for a scenario is covered by SEDEP step six: "Operate Federation".

A selected HLA compliant Federate has a *Simulation Object Model* (SOM) and additional agreements that show what federate-objects can be created, what interactions it can handle and what object-attributes can be set. This information defines the capabilities of a federate and what features can be modified while preparing a scenario.

During the offline phase of preparing a federate for a scenario execution, the capabilities of a federate determine the content of scenario related FC&I settings.
For an object to be present at the start of the scenario the object has to be listed in the FC&I settings. If a selected object has capabilities to be initialized to a certain starting position in the SE, or can be instructed to adopt certain behavior before the scenario execution, then a subset of the object related attributes has to be assigned to the appropriate values and listed in the FC&I settings also. If a federate has to be hosted on a specific platform, the FC&I settings should also contain the appropriate deployment information. A platform can host one or more federates and/or components.

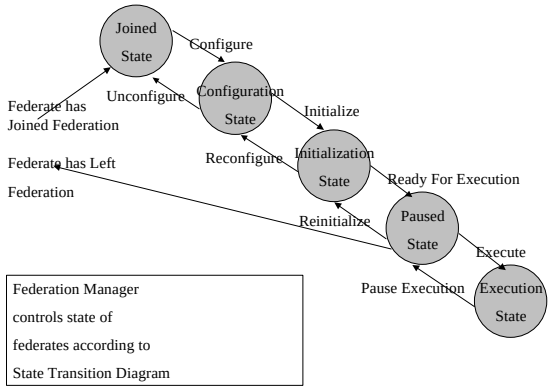

Figure 4: Federate State Diagram

During the online phase of preparing a federate for scenario execution, the offline created FC&I settings are interpreted and deployed accordingly. The activities during preparation are:

1) Select the appropriate FC&I settings according to the selected scenario;
2) Distribution of data and software to the appropriate platforms according to the deployment directions;
3) Bootstrapping of the federates so that they can be accessed remotely;
4) Joining of components to their component-based federate and joining of the federates to the federation;
5) Configuring the scenario objects by creating objects according to the Configuration settings;
6) Initializing the object attributes according to the Initialization settings;
7) Execution of the scenario.

3.1 Tool Requirements

In Euclid 11:13, it was decided to develop a separate supporting tool to handle the activities configuration and initialization during preparation. Requirements for such a tool are:

- Automatic effectuating of configuration and initialization settings, see activity 1, 5 and 6;
- Handle a flexible and extendible FC&I setting format in order to interface to offline preparation;
- Operate from a centralized console;
- Support for component-based federates. Each component-based federate can have its private FC&I settings, that are not visible at federation level. Offer support for deeper nested FC&I settings [MJ3];
- Support for re-configuration and re-initialization (e.g. initialize again and again with different settings).
- It is fundamental within HLA that each federation can handle dedicated data format schemes (in contrast to DIS, where data format was standardized). This HLA agreement covers only communication

during run-time. Within SEDEP we should also allow federation specific data format schemes when dealing with FC&I settings.

Activities that are not covered by the FC&I tool get attention within other work-packages of Euclid11:13.

4. Design Considerations and Decisions

An HLA compliant federate, that is not yet SEDEP enabled, interfaces to the RTI and its SOM has a non empty intersection with its Federation's FOM. A federate's SOM is a set of objects and interactions that are exchanged during run-time. HLA does not specify anything for the initial settings that are to be performed before run-time executions. They are not part of the SOM and can be treated in a dedicated way for each federate. Within SEDEP, it has been decided not only to specify run-time exchanged objects and interactions in a standardized way, but also to specify what objects and interactions are exchanged before entering the "execution state". (i.e. the FC&I settings). Note that the two can overlap (e.g. re-initialization).

The practical use of a proper SOM is two-fold. It advertises the capabilities of a federate, but it also specifies the run-time data exchange (i.e. this is the intersection with its FOM). Both aspects are also important for the FC&I settings. During offline preparation of a scenario, a scenario developer has to deal with the adverted capabilities of a federate, while online the FC&I settings are actually exchanged. Note that an advertisement of capabilities of a candidate federate may include all kinds of functional descriptions, not necessarily needed during execution. An advertisement explains "what" the capabilities are, while a calling interface specifies "how" to invoke it.

4.1 Design Considerations for the FC&I Concept

- Describe both advertisement and interface, using the same description or use different descriptions.
- Describe the FC&I non-runtime settings as an integral part of the SOM in the OMT format, just like any other run-time object and/or interaction. Or choose another dedicated format. Either way, "SEDEP enabled" federates will sill remain HLA compliant.
- The FC&I settings can either be interpreted by each individual federate/component, or this responsibility can be delegated to one dedicated federate (i.e. FC&I tool). This special federate parses all settings for all other federates and it distributes the individual FC&I settings to each appropriate federate/component. See Figure 5 and 6[MJ4].

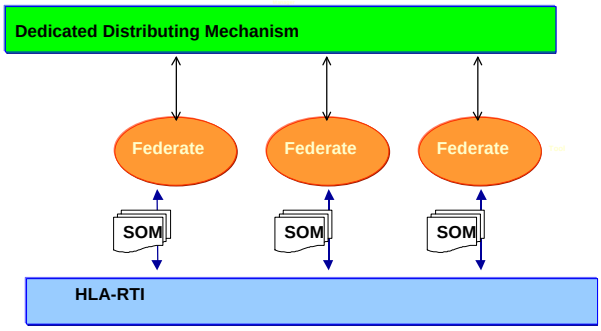

Figure 5: Distribute FC&I Settings (1)

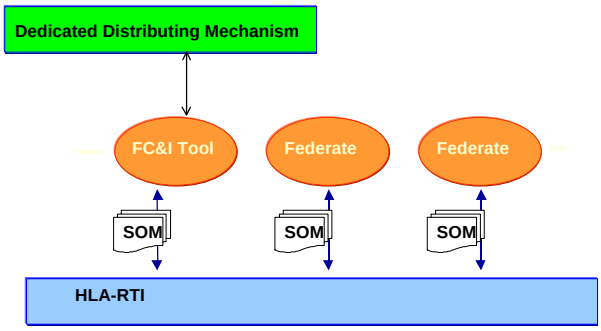

Figure 6: Distribute FC&I Settings (2)

- If the FC&I settings are described as an extension of the SOM, it is obvious that the RTI is used as the distributing middleware. However if another description is chosen, also another dedicated distribution mechanism can be chosen (e.g. file transfer via FTP). In case the RTI is chosen, the RTI should offer transparent channels for complex data types (e.g. an XML file) to distribute settings to a federate, see figure 7. Note that an agreed upon Data Type Definition (DTD) must still be interpreted (and specified) by each individual federate/component.

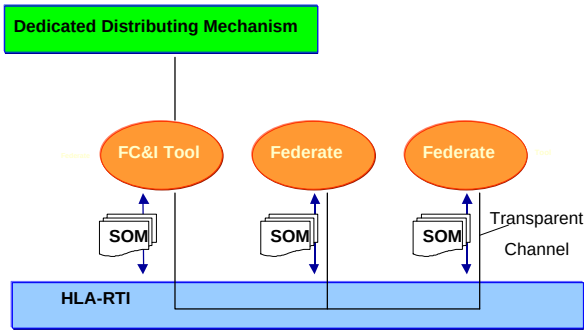

Figure 7: Distribute FC&I Settings (3)

- Given the requirement that a federation can have a dedicated data-formatting scheme. One has to consider what (additional) process steps are responsible for data format conversion in case offline prepared settings do not match.

4.2 Design Decisions

In order to be consistent with the run-time descriptions, we have decided to describe the FC&I settings as an extension of the SOM of a federate in the OMT format. The FC&I settings are composed of objects with attributes and interactions. The declaration of the FC&I settings becomes an integral part of the SOM beside the run-time description.

We did not define a separate file format for the use of advertisement. If a scenario developer has the opportunity to re-use a federate, he must extract from the SOM what FC&I settings can be performed. From the desire to change as little as possible to existing HLA compliant federates we decided to have a separate federate act as an FC&I tool and parse the FC&I settings for all federates, see figure 6. Now only the FC&I tool has to parse and publish the FC&I settings produced by the offline effort. We have chosen an XML/DTD file format to be interpreted by the FC&I tool because this is also used in the OMT IEEE 1516 standard. The FC&I file is to be distributed by another SEDEP tool to the appropriate platform via a file transfer mechanism.

Having chosen the above, it becomes possible to allow the FC&I tool to assume ownership using the standardized negotiated ownership management service of HLA-RTI. The FC&I tool instructs each federate to create its own objects during configuration. In the initialization state however, the FC&I tool temporarily assumes ownership and updates the appropriate FC&I settings. Not using ownership management would require additional dedicated interactions.

Because a federate can be component-based, we decided to 'request' for a state transition before committing to the actual state transition. This extra handshake interaction allows a component-based federate to consult its internal components and match its internal state transition to its external state in a robust way. This responsibility is to be implemented in the Federate Manager.

It was decided that the FC&I should not perform any data format conversion and marshalling of the FC&I settings. Since this aspect is federate-specific for a given scenario, the developer that prepares the scenario should take into account the appropriate data format. This activity is covered in SEDEP step "Implement Federation". Since we would like the tool to be re-used for all kinds of federations, we decided to use a transparent attribute value format (any type) to be assigned to the FC&I settings. Because the FC&I tool does not have to know the federation-specific data format and marshalling scheme, it does not have to interpret the FOM.

We have decided to use two separate states for configuration and initialization. See figure 4. Separating these states promotes the use of re-initialization without having to re-configure. It also helps federate implementers as they can assume that all initial objects have been created. Initialization might refer to an object instance, which could cause problems if the instance does not yet exist.

For a federate to be SEDEP enabled, it is required that it supports the required HLA services (e.g. Ownership Management) as well as responds to the state transitions requested from the Federation Manager (FNM) respectively the Federate Manager (FM). Some dedicated interactions had to be introduced to manage progress during configuration and initialization.

5. Implementation and Testing

The design was documented and reviewed before starting implementation. During the analysis phase Pitch made a mock up of the HMI that is enclosed in the design. See figure 8.

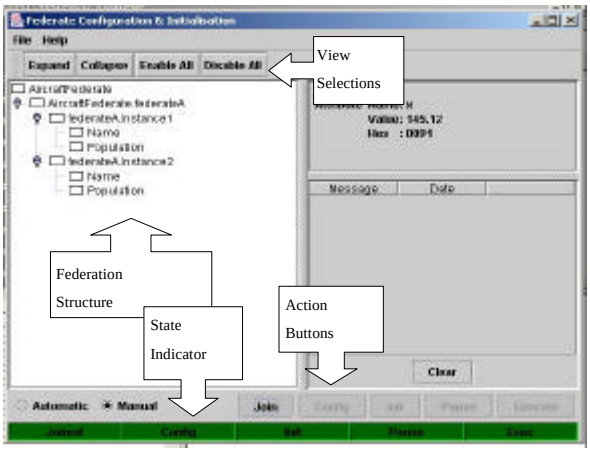

Figure 8: HMI FC&I Tool

The workload was divided such that one party did the development of the tool while the other party did the integration in a test environment. The test environment was built using four federates of which one is component-based. As shown in figure 3. The FC&I tool interfaces to the HLA-RTI without support of TNO's Runtime Communication Infrastructure RCI (See [4]) because of backward compatibility with the underlying HLA-RTI middleware. The test federate/component however, are based on the RCI middleware.

In order to stress the interoperability aspects, the test environment is implemented as a distributed multi-platform site. The FC&I tool instances are running under a Java Virtual Machine and the other federates and components running under IRIX/LINUX.

6. Conclusions

The FC&I prototype tool was successfully completed on the planned date. In a component-based federation, a federate must obey the same interface-format as a component. We have shown successfully that a component implementation can serve as a federate and a federate implementation can serve as a component.

Some of the lessons learned are:

- As mentioned, the FC&I tool can handle various data formats using a transparent data field for assigning FC&I settings. This implies that the data formatting scheme of the FC&I settings is dictated by the FOM development when developing/re-using a federate/component. The FC&I settings are to be generated by scenario tools, developed within SEDEP. One suggestion is to perform data format conversion using a scenario and federation specific conversion utility before the FC&I settings are parsed by the FC&I tool. This utility should parse the federation specific FOM. Another possibility is to standardize a federation generic data formatting scheme and let each individual federate/component perform its data conversion.
- The testing of the FC&I tool showed that some details of the design document needed to be further elaborated. However it never meant that we had to go back to the drawing board.
- During testing the parties were geographically too far apart to meet eye-to-eye. We were forced to explicitly describe the mismatches and shortcomings in the design. This lead to laborious explaining, in writing, about the found issues. However, it forced us to investigate mismatches thoroughly and it was often solved by the next iteration.

7. References

[1] Defense Modeling and Simulation Office (DMSO), "High Level Architecture for Simulations Interface Specification"

[2] Defense Modeling and Simulation Office (DMSO), "High Level Architecture Object Model Template"

[3] Defense Modeling and Simulation Office (DMSO), "Federation Development and Execution Process (FEDEP) Model"

[4] Wim Huiskamp, Henk Janssen, Hans Jense, "An HLA based Flight Simulation Architecture", Proc. AIAA Modeling and Simulation Technologies Conference, Denver, Colorado, USA, August 2000 (AAIA-2000-4401)

[6] Keith Ford: "Euclid RTP11.13 Realising the Potential of Networked Simulation in Europe", 01E-SIW-011, European Simulation Interoperability Workshop 2001

[7] Keith Ford: "The Euclid 11.13 SE Development Environment", 02E-SIW-007, European Simulation Interoperability Workshop 2002

Author Biographies

JAN JELLE BOOMGAARDT is a research scientist in the Command & Control and Simulation Division at TNO-FEL. He holds an M.Sc. degree in Software Engineering from Twente University of Technology, The Netherlands. He works in the field of High Performance Computing and Networking and specialises in design and implementation of distributed computer architectures.

MAGNUS JOHANSSON is a software developer at Pitch AB. He holds a MSc in Computer Science from Linköping Institute of Technology, Sweden. He works in the field of HLA federate and federation development and specialises in design and automated development of HLA compliant Computer Generated Forces.

Page: 3

[MJ1]remove space

Page: 3

[MJ2]remove one space

Page: 4

[MJ3]remove

Page: 5

[MJ4]remove