IEEE STANDARDS ASSOCIATION

IEEE Recommended Practice for Distributed Simulation Engineering and Execution Process (DSEEP)

IEEE Computer Society

Sponsored by the
Simulation Interoperability Standards Organization (SISO)

IEEE
3 Park Avenue
New York, NY 10016-5997
USA

24 January 2011

IEEE Std 1730™-2010
(Revision of
IEEE Std 1516.3™-2003)

IEEE Std 1730™-2010
(Revision of
IEEE Std 1516.3™-2003)

IEEE Recommended Practice for Distributed Simulation Engineering and Execution Process (DSEEP)

Sponsor

Simulation Interoperability Standards Organization (SISO)
of the
IEEE Computer Society

Approved 30 September 2010
IEEE-SA Standards Board

Approved 10 June 2011
American National Standards Institute

Abstract: Recommended practice for Distributed Simulation Engineering and Execution Process (DSEEP) is described. The DSEEP is intended as a high-level process framework into which the lower-level systems engineering practices native to any distributed simulation user can be easily integrated. Simulation architectures include Distributed Interactive Simulation (DIS), High Level Architecture (HLA), and Test and Training Enabling Architecture (TENA).

Keywords: DIS, distributed simulation, engineering methodology, HLA, IEEE 1730, M&S, system process, TENA

The Institute of Electrical and Electronics Engineers, Inc.
3 Park Avenue, New York, NY 10016-5997, USA

PDF: ISBN 978-0-7381-6468-7 STD97022
Print: ISBN 978-0-7381-6469-4 STDPD97022

IEEE Standards documents are developed within the IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Association (IEEE-SA) Standards Board. The IEEE develops its standards through a consensus development process, approved by the American National Standards Institute, which brings together volunteers representing varied viewpoints and interests to achieve the final product. Volunteers are not necessarily members of the Institute and serve without compensation. While the IEEE administers the process and establishes rules to promote fairness in the consensus development process, the IEEE does not independently evaluate, test, or verify the accuracy of any of the information or the soundness of any judgments contained in its standards.

Use of an IEEE Standard is wholly voluntary. The IEEE disclaims liability for any personal injury, property or other damage, of any nature whatsoever, whether special, indirect, consequential, or compensatory, directly or indirectly resulting from the publication, use of, or reliance upon this, or any other IEEE Standard document.

The IEEE does not warrant or represent the accuracy or content of the material contained herein, and expressly disclaims any express or implied warranty, including any implied warranty of merchantability or fitness for a specific purpose, or that the use of the material contained herein is free from patent infringement. IEEE Standards documents are supplied "**AS IS**."

The existence of an IEEE Standard does not imply that there are no other ways to produce, test, measure, purchase, market, or provide other goods and services related to the scope of the IEEE Standard. Furthermore, the viewpoint expressed at the time a standard is approved and issued is subject to change brought about through developments in the state of the art and comments received from users of the standard. Every IEEE Standard is subjected to review at least every five years for revision or reaffirmation, or every ten years for stabilization. When a document is more than five years old and has not been reaffirmed, or more than ten years old and has not been stabilized, it is reasonable to conclude that its contents, although still of some value, do not wholly reflect the present state of the art. Users are cautioned to check to determine that they have the latest edition of any IEEE Standard.

In publishing and making this document available, the IEEE is not suggesting or rendering professional or other services for, or on behalf of, any person or entity. Nor is the IEEE undertaking to perform any duty owed by any other person or entity to another. Any person utilizing this, and any other IEEE Standards document, should rely upon his or her independent judgment in the exercise of reasonable care in any given circumstances or, as appropriate, seek the advice of a competent professional in determining the appropriateness of a given IEEE standard.

Interpretations: Occasionally questions may arise regarding the meaning of portions of standards as they relate to specific applications. When the need for interpretations is brought to the attention of IEEE, the Institute will initiate action to prepare appropriate responses. Since IEEE Standards represent a consensus of concerned interests, it is important to ensure that any interpretation has also received the concurrence of a balance of interests. For this reason, IEEE and the members of its societies and Standards Coordinating Committees are not able to provide an instant response to interpretation requests except in those cases where the matter has previously received formal consideration. A statement, written or oral, that is not processed in accordance with the IEEE-SA Standards Board Operations Manual shall not be considered the official position of IEEE or any of its committees and shall not be considered to be, nor be relied upon as, a formal interpretation of the IEEE. At lectures, symposia, seminars, or educational courses, an individual presenting information on IEEE standards shall make it clear that his or her views should be considered the personal views of that individual rather than the formal position, explanation, or interpretation of the IEEE.

Comments for revision of IEEE Standards are welcome from any interested party, regardless of membership affiliation with IEEE. Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments. Recommendations to change the status of a stabilized standard should include a rationale as to why a revision or withdrawal is required. Comments and recommendations on standards, and requests for interpretations should be addressed to:

Secretary, IEEE-SA Standards Board
445 Hoes Lane
Piscataway, NJ 08854
USA

Introduction

This introduction is not part of IEEE Std 1730-2010, IEEE Recommended Practice for Distributed Simulation Engineering and Execution Process (DSEEP).

Modeling and simulation (M&S) has long been an enabler of many key activities across a broad spectrum of different user communities. Such activities include the use of M&S to support the training of personnel, the test and evaluation of new systems and capabilities, analysis of logistics issues, and much more. At its core, M&S is all about the reduction of risk. That is, by providing abstract representations of highly complex systems along with the operational environment in which the system exists, users can gain an understanding of the performance and behavior of the system that would not be achievable otherwise. Thus, the insight gained through careful and thoughtful application of M&S assets provides an invaluable means of making wise decisions about how the system should be designed, developed, and employed.

There is a wide variety of models and simulations in use today. Such tools are generally characterized by their general class (live, virtual, or constructive) and by the level of detail that they support. While most M&S tools are designed as stand-alone applications, recent advances in software and networking technology has led to much greater use of distributed simulation environments. In such environments, the unique functionalities offered by a selected set of disparate models and simulations are combined and linked using high-speed networks and advanced simulation services to create an integrated system that appears to users to be a single, very powerful simulation.

There are many different approaches to building distributed simulation environments that rely on a wide variety of different protocols, middleware products, and data management schemes. The choice of which systems engineering methodology works best in any given situation depends on several factors, such as the class of application (e.g., analysis, test, training), the types of M&S assets being linked together, *verification, validation, and accreditation/acceptance* (VV&A) and security concerns, cost/schedule constraints, and even the background and experience of the development/integration team. While considerable experimentation within the distributed simulation community has driven the current wide range of modern, innovative distributed simulation development and execution methodologies and supporting implementation mechanisms in use today, the proliferation of such methodologies has made it more difficult to transition M&S assets (including the people that support them) from one user domain to another, and in some cases, has become a barrier to interoperability across the environments supported by different user domains employing different methodologies.

One means of addressing these problems is to compare the various approaches to distributed simulation environments, and look for the common denominators. That is, determine if there are certain activities and tasks that are common across distributed simulation communities and capture these within a common, unified process model. A common process would be quite valuable for several reasons:

— There would be a common basis for comparison of different domain-specific methodologies. That is, by identifying how the various methodologies are the same, it also identifies how they are different, and thus facilitates the understanding of how and when one methodology should be used versus another.

— It would provide a common framework for different communities to describe their native systems engineering practices. That is, by describing their engineering practices within the context of the generalized activities and tasks in the common process model, they can better communicate with and understand the approaches used by other communities that use this same process framework, and help them to drive toward commonality in development approaches whenever possible.

— It would provide a common language for building mixed-architecture simulation environments. That is, in circumstances in which more than one architecture (e.g., HLA, DIS, TENA) is required to meet all of the technical, cost, and schedule constraints associated with the application, a

common process model can help to bridge the communications barriers that frequently occur between the users of different simulation architectures, and helps to identify and establish the necessary agreements that must be made for such mixed-architecture environments to operate properly.

— It would provide an effective "first read" to people that are new to the distributed simulation community that want to learn and understand the high-level view of the current best practices in the distributed simulation community before immersing themselves in the specific systems engineering approach used within their own user domain.

This document describes the recommended practice for DSEEP. The purpose of this document is to describe a generalized process for building and executing distributed simulation environments. It is not intended to replace the existing management and systems design/development methodologies of user organizations, but rather to provide a high-level framework for simulation environment construction and execution into which lower-level systems engineering practices native to each individual application area can be easily integrated. In addition, the DSEEP is not intended to be prescriptive, in that it does not specify a "one size fits all" process for all users of distributed simulation. Rather, the DSEEP defines a generic, common-sense systems engineering methodology that can and should be tailored to meet the needs of individual applications.

Although every distributed simulation application requires a basic agreement among all member application representatives as to the systems engineering approach that will be used to develop and execute the simulation environment, there can be significant variability in the degree of formality defined in the chosen process. The primary driver for how much formality is required is the size and complexity of the application. For example, in large complex applications with many distributed member applications, project control requirements generally dictate the need for a rigidly defined, highly structured process to facilitate proper communication and coordination among all team members. In such environments, requirements and associated schedules for delivery of supporting products are generally very explicit, as is the content and format for documentation of these products. In smaller or less complex applications, a less structured process with fewer constraints on the types, formats, and content of supporting products may be perfectly reasonable and may have certain efficiency advantages as compared to a more formalized process.

Other secondary factors may also drive the process selected for a specific application. For instance, some communities may have documentation requirements that are unique to their application area. In this case, the activities required to produce these products must be accounted for in the overall process. The reuse potential of these and other required products may also influence the nature and formality of the activities that produce them. Another factor is the availability of reusable products and persistent development/integration teams as opportunities for shortcuts, whereby it may be possible to identify and take advantage of a more streamlined, efficient development/integration process. Finally, practical resource constraints (e.g., cost, schedule) may dictate how certain activities are performed and how the associated products are produced and documented.

In summary, it is recognized that the needs and requirements of the distributed simulation community are quite diverse. The DSEEP is offered to the distributed simulation community as a means of identifying today's best practices for the development and execution of simulation environments, and for identifying and addressing the various technical and managerial issues associated with implementation of these practices. The DSEEP can and should be tailored as appropriate to address the unique issues, requirements, and practical constraints of each individual application. It is expected that this framework will provide a viable foundation for all distributed simulation applications and will assist the users in defining the specific tasks and activities necessary to support their particular needs.

Notice to users

Laws and regulations

Users of these documents should consult all applicable laws and regulations. Compliance with the provisions of this standard does not imply compliance to any applicable regulatory requirements. Implementers of the standard are responsible for observing or referring to the applicable regulatory requirements. IEEE does not, by the publication of its standards, intend to urge action that is not in compliance with applicable laws, and these documents may not be construed as doing so.

Copyrights

Updating of IEEE documents

Users of IEEE standards should be aware that these documents may be superseded at any time by the issuance of new editions or may be amended from time to time through the issuance of amendments, corrigenda, or errata. An official IEEE document at any point in time consists of the current edition of the document together with any amendments, corrigenda, or errata then in effect. In order to determine whether a given document is the current edition and whether it has been amended through the issuance of amendments, corrigenda, or errata, visit the IEEE Standards Association web site at http://ieeexplore.ieee.org/xpl/standards.jsp, or contact the IEEE at the address listed previously.

For more information about the IEEE Standards Association or the IEEE standards development process, visit the IEEE-SA web site at http://standards.ieee.org.

Errata

Errata, if any, for this and all other standards can be accessed at the following URL: http://standards.ieee.org/reading/ieee/updates/errata/index.html. Users are encouraged to check this URL for errata periodically.

Interpretations

Current interpretations can be accessed at the following URL: http://standards.ieee.org/reading/ieee/interp/index.html.

Patents

Attention is called to the possibility that implementation of this recommended practice may require use of subject matter covered by patent rights. By publication of this recommended practice, no position is taken with respect to the existence or validity of any patent rights in connection therewith. The IEEE is not responsible for identifying Essential Patent Claims for which a license may be required, for conducting inquiries into the legal validity or scope of Patents Claims or determining whether any licensing terms or conditions provided in connection with submission of a Letter of Assurance, if any, or in any licensing agreements are reasonable or non-discriminatory. Users of this recommended practice are expressly advised that determination of the validity of any patent rights, and the risk of infringement of such rights, is entirely their own responsibility. Further information may be obtained from the IEEE Standards Association.

Participants

At the time this recommended practice was submitted to the IEEE-SA Standards Board for approval, the DSEEP PDG Working Group had the following membership:

Robert Lutz, *Chair*
Katherine L. Morse, *Co-Vice Chair*
Jean-Louis Igarza, *Co-Vice Chair*
Jake Borah, *Secretary*
Paul Gustavson, *Drafting Group Editor*

Richard Alisch
Joanne Atherton
Jane Bachman
William Beavin
Curtis Blais
Derrick Briscoe
Arthur Brooks
Bang Choon
Dannie Cutts*
Raymond Dean
Klaas Jan de Kraker
Bradford Dillman
Hoang Doan
Arren Dorman
Mark Dumble
Sarah Epps*
Joey Fann
Keith Ford*
Frank Hill
Jim Hollenbach
Wim Huiskamp
Stephen Jones
Fredrik Jonsson
Phil Landweer
Jeremy Lanman
Jennifer Lewis
Mike Lightner
Reed Little*
Staffan Löf
Bjorn Lofstrand*
Max Lorenzo
Paul Lowe*
Van Lowe
Farid Mamaghani
Lance Marrou
James McCall*
Ted Miller
Bjorn Möller*
Scott Newman
Colin Petford
Mikel Petty
Edward Powell
Mick Priestley
Laurent Prignac
Richard Reading*
Lawrence Root
Peter Ross
Chris Rouget*
Roy Scrudder
Graham Shanks
Steven Sheasby
Petr Shlyaev
Tom Shook
Robert Siegfried
Marcy Stutzman
David Taylor
Cam Tran*
Tom van den Berg*
Rene Verhage
Daniel Verret
Douglas Wakefield
Lucien Zalcman

*Represents member of the PDG Drafting Group.

The following members of the individual balloting committee voted on this recommended practice. Balloters may have voted for approval, disapproval, or abstention.

Jane Bachman
Tom Berg
Mitchell Bonnett
Jack Borah
Juan Carreon
Keith Chow
Dannie Cutts
Sarah Epps
Randall Groves
Paul Gustavson
Werner Hoelzl
Rameshchandra Ketharaju
Murray Little
Paul Lowe
Robert Lutz
Edward McCall
James McCall
Bjorn Möller
Katherine L. Morse
Thomas Mullins
Richard Reading
Peter Ryan
Randy Saunders
Bartien Sayogo
Stephen Schwarm
Roy Scrudder
Steven Sheasby
Gil Shultz
James Smith
Thomas Starai
Walter Struppler
Marcy Stutzman
Michael Swearingen
Cam Tran
Paul Work

When the IEEE-SA Standards Board approved this recommended practice on 30 September 2010, it had the following membership:

Robert M. Grow, *Chair*
Richard H. Hulett, *Vice Chair*
Steve M. Mills, *Past Chair*
Judith Gorman, *Secretary*

Karen Bartleson
Victor Berman
Ted Burse
Clint Chaplin
Andy Drozd
Alexander Gelman
Jim Hughes
Young Kyun Kim
Joseph L. Koepfinger*
John Kulick
David J. Law
Hung Ling
Oleg Logvinov
Ted Olsen
Ronald C. Petersen
Thomas Prevost
Jon Walter Rosdahl
Sam Sciacca
Mike Seavey
Curtis Siller
Don Wright

*Member Emeritus

Also included are the following nonvoting IEEE-SA Standards Board liaisons:

Satish K. Aggarwal, *NRC Representative*
Richard DeBlasio, *DOE Representative*
Michael Janezic, *NIST Representative*

Don Messina
IEEE Standards Program Manager, Document Development

Michael D. Kipness
IEEE Standards Program Manager, Technical Program Development

Contents

1. Overview 1

2. Definitions, acronyms, and abbreviations 1
2.1 Definitions 1
2.2 Acronyms and abbreviations 2

3. DSEEP: Top-level process flow view 3

4. DSEEP: Detailed product flow view 5
4.1 Step 1: Define simulation environment objectives 9
4.2 Step 2: Perform conceptual analysis 13
4.3 Step 3: Design simulation environment 17
4.4 Step 4: Develop simulation environment 24
4.5 Step 5: Integrate and test simulation environment 29
4.6 Step 6: Execute simulation 33
4.7 Step 7: Analyze data and evaluate results 36

5. Conclusion 38

Annex A (normative) High level architecture process overlay to the DSEEP 39

Annex B (normative) Distributed Interactive Simulation process overlay to the DSEEP 45

Annex C (normative) Test and Training Enabling Architecture process overlay to the DSEEP 55

Annex D (informative) Bibliography 66

IEEE Recommended Practice for Distributed Simulation Engineering and Execution Process (DSEEP)

1. Overview

This recommended practice defines the processes and procedures that should be followed by users of distributed simulations to develop and execute their simulations; it is intended as a higher-level framework into which low-level management and systems engineering practices native to user organizations can be integrated and tailored for specific uses.

2. Definitions, acronyms, and abbreviations

2.1 Definitions

For the purposes of this document, the following terms and definitions apply. *The IEEE Standards Dictionary: Glossary of Terms & Definitions* should be consulted for terms not defined in this clause.[1]

conceptual model: An abstraction of what is intended to be represented within a simulation environment, which serves as a frame of reference for communicating simulation-neutral views of important entities and their key actions and interactions. The conceptual model describes what the simulation environment will represent, the assumptions limiting those representations, and other capabilities needed to satisfy the user's requirements. Conceptual models are bridges between the real world, requirements, and design.

[1] *The IEEE Standards Dictionary: Glossary of Terms & Definitions* is available at http://shop.ieee.org/.

member application: An application that is serving some defined role within a simulation environment. This can include live, virtual, or constructive simulation assets, or can be supporting utility programs such as data loggers or visualization tools.

objective: The desired goals and results of the activity to be conducted in the distributed simulation environment expressed in terms relevant to the organization(s) involved.

repository: A central place where modeling and simulation (M&S) resources may be cataloged, stored or accessed.

requirement: A statement identifying an unambiguous and testable characteristic, constraint, process or product of an intended simulation environment.

simulation data exchange model (SDEM): A specification defining the information exchanged at runtime to achieve a given set of simulation objectives. This includes class relationships, data structures, parameters, and other relevant information.

simulation environment: A named set of member applications along with a common simulation data exchange model (SDEM) and set of agreements that are used as a whole to achieve some specific objective.

scenario: (A) Description of an exercise. It is part of the session database that configures the units and platforms and places them in specific locations with specific missions. **(B)** An initial set of conditions and time line of significant events imposed on trainees or systems to achieve exercise objectives.

NOTE—For definition **(A)**, see Tucker [B4].[2, 3]

2.2 Acronyms and abbreviations

BOM	base object model
CASE	computer aided software engineering
COTS	commercial off-the-shelf
CPU	central processing unit
DIS	Distributed Interactive Simulation
DSEEP	Distributed Simulation Engineering and Execution Process
FOM	federation object model
GOTS	government off-the-shelf
HLA	High Level Architecture
IEEE	Institute for Electrical and Electronics Engineers
LAN	local area network
LROM	logical range object model
M&S	modeling and simulation
MOE	measure of effectiveness
MOP	measure of performance

[2] The numbers in brackets correspond to those of the bibliography in Annex D.
[3] Notes in text, tables, and figures of a standard are given for information only and do not contain requirements needed to implement this standard.

PDU	protocol data unit
SDEM	simulation data exchange model
SOM	simulation object model
RTI	runtime infrastructure
TENA	Test and Training Enabling Architecture
VV&A	verification, validation, and accreditation/acceptance
WAN	wide area network

3. DSEEP: Top-level process flow view

Because of the great diversity among the needs of the distributed simulation user community, there is a recognized desire to avoid mandating unnecessary constraints on how such environments are developed and executed, and generally to provide a high degree of flexibility with regard to how supporting processes are structured. For instance, the types and sequence of low-level activities required to develop and execute distributed analysis-oriented simulation environments is likely to be quite different from those required to develop and execute distributed training exercises. However, at a more abstract level, it is possible to identify a sequence of seven very basic steps that all distributed simulation applications will need to follow to develop and execute their simulation environments. Figure 1 illustrates each of these steps and is summarized below:

— *Step 1*: *Define simulation environment objectives*. The user, the sponsor, and the development/integration team define and agree on a set of objectives and document what must be accomplished to achieve those objectives.

— *Step 2*: *Perform conceptual analysis*. The development/integration team performs scenario development and conceptual modeling, and develops the simulation environment requirements based upon the characteristics of the problem space.

— *Step 3*: *Design simulation environment*. Existing member applications that are suitable for reuse are identified, design activities for member application modifications and/or new member applications are performed, required functionalities are allocated to the member application representatives, and a plan is developed for the development and implementation of the simulation environment.

— *Step 4*: *Develop simulation environment*. The simulation data exchange model (SDEM) is developed, simulation environment agreements are established, and new member applications and/or modifications to existing member applications are implemented.

— *Step 5*: *Integrate and test simulation environment*. Integration activities are performed, and testing is conducted to verify that interoperability requirements are being met.

— *Step 6: Execute simulation*. The simulation is executed and the output data from the execution is preprocessed.

— *Step 7: Analyze data and evaluate results.* The output data from the execution is analyzed and evaluated, and results are reported back to the user/sponsor.

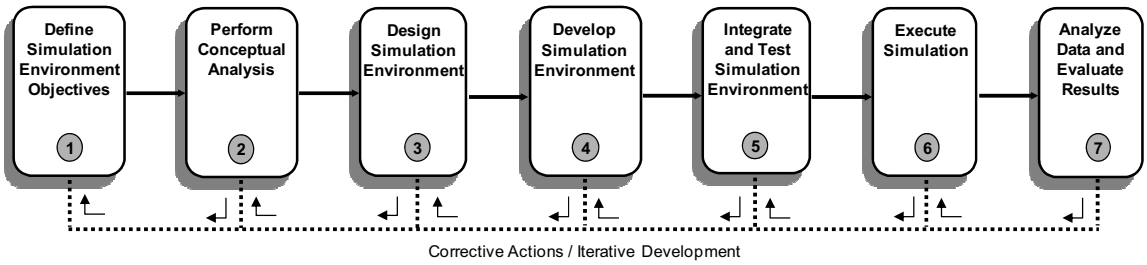

Figure 1—Distributed Simulation Engineering and Execution Process (DSEEP), top-level process flow view

Since this seven-step process can be implemented in many different ways depending on the nature of the application, it follows that the time and effort required to build and execute a simulation environment can also vary significantly. For instance, it may take a development/integration team several iterations spanning several weeks to months to fully define the real-world domain of interest for large, complex applications. In smaller, relatively simple applications, the same activity could potentially be conducted in a day or less. Differences in the degree of formality desired in the process can also lead to varying requirements for supporting resources.

Within any step of the DSEEP, it may become apparent that the simulation approach is unlikely to ultimately provide sufficiently reliable output data for meaningful analysis to be carried out and therefore that the process should be prematurely terminated. There could be many reasons for this, for example, cost, timescales, lack of viable revised objectives, or non-availability of appropriate member applications. In order to identify such a situation as soon as possible, each major step of the DSEEP should include an assessment as to whether the process should continue or be terminated. Any reason for early termination should be clearly documented.

Personnel requirements can also vary greatly depending on the scope of the distributed simulation environment. In some situations, highly integrated teams composed of several individuals may be needed to perform a single role in a large, complex simulation environment, while a single individual may perform multiple roles in smaller simulation environments. Each distributed simulation user will define their roles according to the needs of their application, the domain they support, and the function performed. Some roles are unique to a single activity in the process, while others are more pervasive throughout the process. Since the applicability of a given role (as well as the set of activities it spans) varies from simulation environment to simulation environment, the activities described in this document specify the roles of individuals only in generic terms. However, best practices recognize some roles that will be prevalent in all distributed simulation activities, regardless of their titles, as follows:

— **User/sponsor:** The person, agency, or organization who determines the need and scope of a distributed simulation exercise or event, and establishes the funding and other required resources. The user/sponsor also approves the participants, objectives, requirements, and specifications. The user/sponsor appoints the simulation environment manager and verification, validation, and accreditation/acceptance (VV&A) agents.

— **Simulation environment manager:** The person responsible for creating the simulation environment, executing the event(s) in the simulation environment, and conducting the post-event activities. The simulation environment manager coordinates with the VV&A agent(s) during these tasks and then reports the results of the event(s) to the user/sponsor.

— **Development/integration team:** The team developing a simulation environment, integrating member applications and systems into the simulation environment, planning for the all aspects of the process, and responsible for ensuring simulations are compliant with simulation environment agreements.

— **Verification and validation (V&V) agent(s):** The person, agency, or organization responsible for verifying and validating member applications or simulation environment.

— **Accreditation/acceptance agent(s):** The person, agency, or organization that accredits member applications or simulation environments for use and reuse for specific purposes or categories of purposes; responsible for certifying that a simulation environment has been verified and validated; and authorizes the use of the simulation environment for its intended use.

A major source of variation in how the seven-step process is implemented relates to the degree of reuse of existing products. In some cases, no previous work may exist; therefore a new simulation environment may need to be developed based on the defined requirements. In other cases, user communities with established, persistent simulation environments will receive additional requirements. In this circumstance, the users can choose to reuse previous work (e.g., member applications, SDEMs, planning documents) either in part or whole, along with the products of new developmental activities. When an appropriate management structure exists to facilitate this type of development environment, significant savings can be achieved in both cost and development time.

The remainder of this document describes a structured, systems engineering approach to simulation environment development and execution known as the *Distributed Simulation Engineering and Execution Process* (DSEEP). The seven-step process provides a top-level, process-flow view of the major steps of DSEEP. In the detailed product-flow view, each step is further decomposed into a set of interrelated lower-level activities and supporting information resources. Since the needs of distributed simulation users range from "first use" applications to experienced users, the DSEEP makes no assumptions about the existence of an established core set of member applications or the up-front availability of reusable products. Further, there are no assumptions about the size of the simulation environment, as even users and developers of stand-alone (non-distributed) simulations can benefit from the guidance provided in this document. Although the intention is to define a comprehensive, generalized framework for the construction and execution of distributed simulation environments, it is important to recognize that users of this process model will normally need to adjust and modify the DSEEP as appropriate to address the unique requirements and constraints of their particular application area.

4. DSEEP: Detailed product flow view

The DSEEP describes a high-level framework for the development and execution of distributed simulation environments. The intent of the DSEEP is to specify a set of guidelines for the development and execution of these environments that stakeholders can leverage to achieve the needs of their application.

A detailed, product-flow view of the DSEEP is provided in Figure 2. This view illustrates the flow of information across the seven process steps identified in Figure 1. Data flow diagram notation is used in Figure 2 and throughout this document to represent activities (rounded rectangles), data stores (cylinders), and information flows and products (arrows) (see Scrudder et al. [B1]).

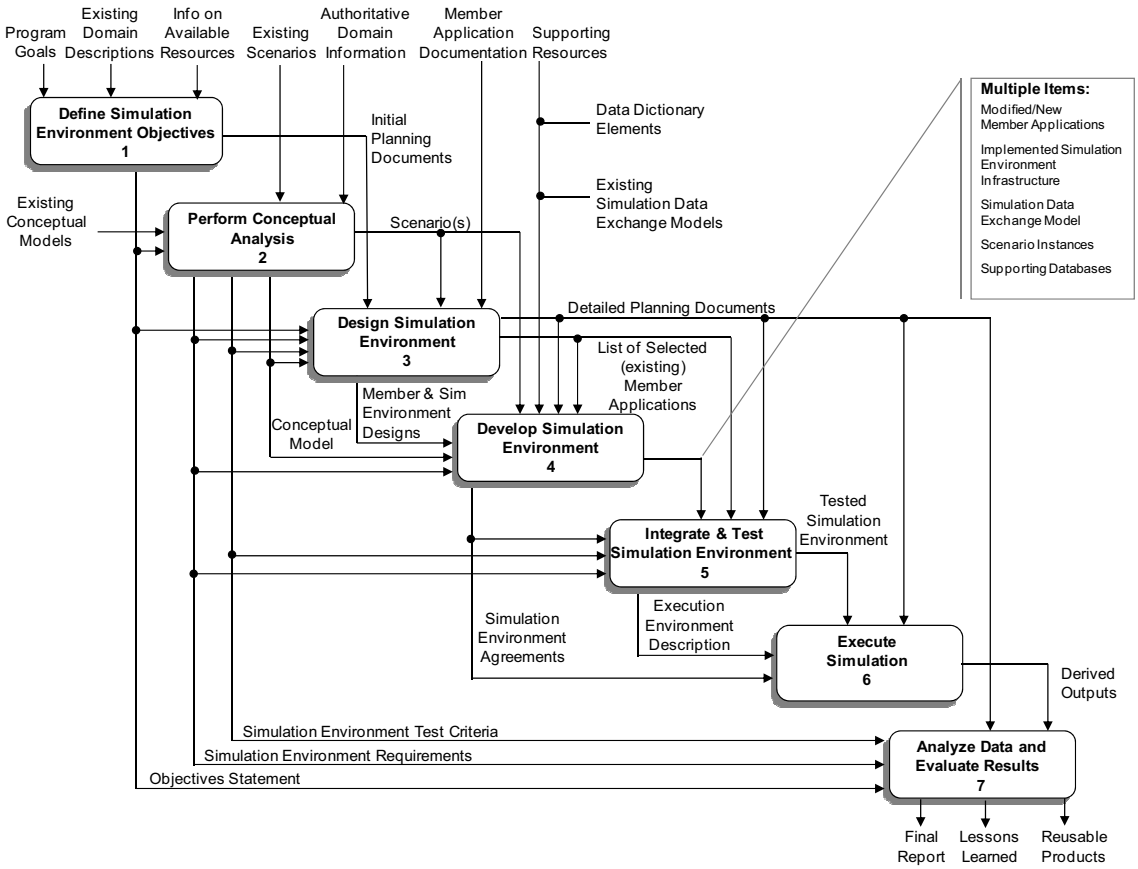

Figure 2—Distributed Simulation Engineering and Execution Process (DSEEP), detailed product flow view

The following subclauses describe the lower-level activities associated with each of the seven major development and execution steps. A tabular view of the activities inherent to each major step is provided in Table 1. Activity descriptions and potential inputs and outputs for the activity and a representative list of recommended tasks are provided. Graphical illustrations of the interrelationships among the activities within each step are also provided. Whenever outputs from one DSEEP activity represent a major input to one or more other activities, the arrow labels explicitly identify the activities that use these outputs. The arrow labels also identify the activities that produce inputs. However, there is a presumption embodied within the DSEEP that once a DSEEP product has been created, it will be available for all subsequent activities, even though the product may not be identified as an input in the activity description. Additionally, once a product is developed, the product may be modified or updated by subsequent activities without such modifications being explicitly identified either as a task or output. This is the natural cause and effect of iterative and incremental development, which is supported by the DSEEP. Input and output arrows without activity number labels are those in which the information originates from outside or is used outside the scope of the DSEEP.

Table 1—Tabular view of the DSEEP

Step	(1) Define simulation environment objectives	(2) Perform conceptual analysis	(3) Design simulation environment	(4) Develop simulation environment	(5) Integrate and test simulation environment	(6) Execute simulation	(7) Analyze data and evaluate results
Activities	Identify user/ sponsor needs Develop objectives Conduct initial planning	Develop scenario Develop conceptual model Develop simulation environment requirements	Select member applications Design simulation environment Prepare detailed plan	Develop simulation data exchange model Establish simulation environment agreements Implement member application designs Implement simulation environment infrastructure	Plan execution Integrate simulation environment Test simulation environment	Execute simulation Prepare simulation environment outputs	Analyze data Evaluate and feedback results

Although many of the activities represented in the DSEEP diagram and tabular view appear highly sequential, the intention is not to suggest a strict waterfall approach to development and execution. Rather, this process illustration is simply intended to highlight the major activities that occur during development and execution and approximately when such activities are first initiated relative to other development activities. In fact, experience has shown that many of the activities shown in Figure 2 as sequential are actually cyclic and/or concurrent, as was indicated earlier in Figure 1 via the dotted feedback arrows. In general, the lower level activities will be conducted in the order implied by the information flow between these activities, but the actual timing of these activities is not absolute and is dependent upon the systems engineering practices actually being used.

In addition to the DSEEP products, there are recommended simulation management planning areas for which persistent documentation is key when using incremental development. A summarized list of these planning areas and resulting documents that are recommended follows:

Development and execution planning. The resulting document from this activity identifies the overall plan of action and milestones for the simulation development and execution. There can be multiple views created to depict the schedule of activities, including detailed task and milestone identification for a particular simulation build.

Verification and validation (V&V) planning. The resulting document from this activity identifies the methodology and guidelines for verifying and validating the simulation environment.

Test planning. The resulting document from this activity identifies the methodology and guidelines for executing and evaluating the simulation environment via a formal test. It should include the test criteria

imposed on the simulation environment as well as specific test requirements for participating member applications.

Configuration management planning. The resulting document from this activity identifies the methodology and guidelines for establishing and managing configuration baselines. It should indicate the process by which design changes to the simulation environment can be made and the version control utilized by participating member applications.

Security planning. The resulting document from this activity identifies the level of security imposed on the development and execution of the simulation environment. It should also identify the specific security requirements required by participating member applications and possible designated approval authority required by participating member applications.

Integration planning. The resulting document from this activity identifies the methodology and guidance for integration throughout the simulation development.

Data management planning. The resulting document from this activity identifies the strategy for data collection, management, and analysis of the simulation environment as well as the member applications.

The plan information identified above may be captured in distinct documents or folded into other documents that are being mapped in the process, and potentially stored in appropriate repositories. It should be noted that more than one individual may be required to populate and mature these documents as there may be specific program expertise or simulation expertise required. A suggestion for management of the planning documentation is to nominate individuals to maintain and update the documents on a periodic basis or as new information is obtained.

Throughout the DSEEP, there will be areas in which additional information could provide useful guidance in the management, maintenance, and development of a simulation environment. This information should be produced as required in a format that complements the documentation structure being used in the specific process implementation. The information can be captured in distinct documents or folded in to other documents that are being managed in the process. Areas in which additional information may be captured include the following:

Management tools. Identify what management tools have been selected to support the DSEEP activities (such as scenario development, requirements, conceptual analysis, VV&A, and configuration management) and how they are being utilized.

Reusable products. Identify the reusable products that have been produced or consumed in the DSEEP activities, such as designs, specifications, source code, documentation, test suites, manuals, procedures, etc.

Simulation environment support tools. Identify what support tools have been selected for the simulation environment (such as data reduction tools, visualization, and simulation environment manager) and how those tools are being utilized in the simulation environment.

Users of the DSEEP should be aware that the activities described in this document, while being generally applicable to most simulation environments, are intended to be tailored to meet the needs of each individual application. For example, DSEEP users should not feel constrained by the products explicitly identified in this document, but rather should produce whatever additional documentation is necessary to support their application. Specific tailorings of the DSEEP based on the native systems engineering processes inherent to certain widely-used distributed simulation architectures [Distributed Interactive Simulation (DIS), High Level Architecture (HLA), and Test and Training Enabling Architecture (TENA)] are included as annexes to this document. In general, the recommended practices provided in this document should be used as a starting point for developing the most appropriate approach to development and execution for the intended application.

In the subclauses that follow, the description of activities for each DSEEP step is provided and includes a supporting data flow diagram. Each data flow diagram identifies at least one individual activity that is labeled by a number designation (X.Y) to show traceability between the step (X) and the activity (Y) and, in the seven-step process, to which the activity is associated. The activity number in these diagrams is intended primarily as an identifier and does not prescribe a particular ordering required for completing a DSEEP step.

4.1 Step 1: Define simulation environment objectives

The purpose of Step 1 of the DSEEP is to define and document a set of needs that are to be addressed through the development and execution of a simulation environment and to transform these needs into a more detailed list of specific objectives for that environment.

Figure 3 illustrates the key activities in this step of the DSEEP. The subclauses that follow describe each of these activities in detail.

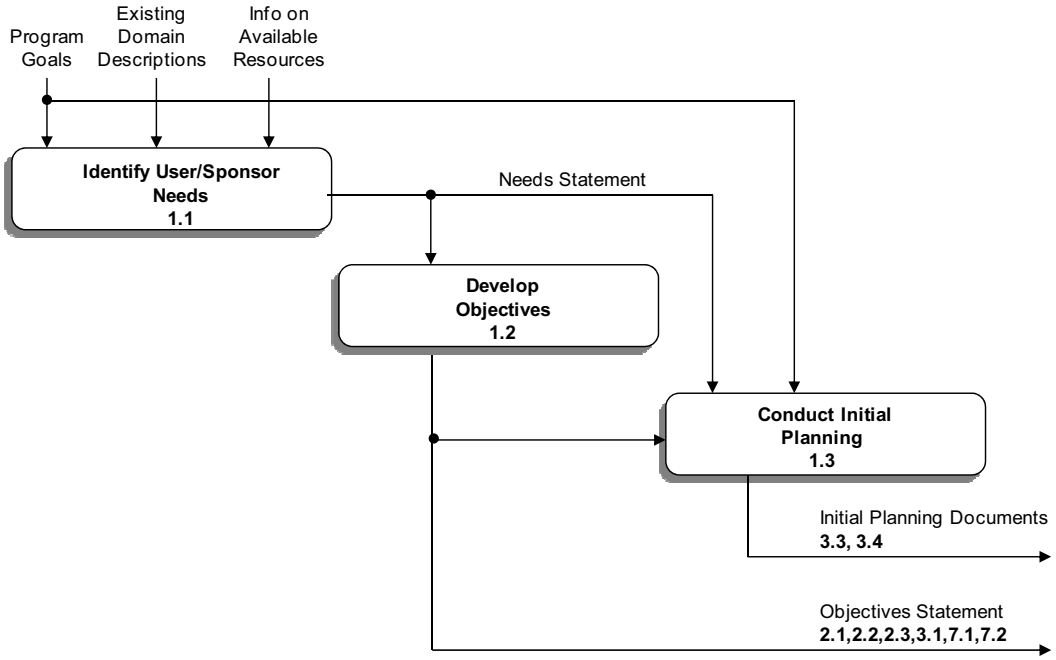

Figure 3—Define simulation environment objectives (Step 1)

4.1.1 Activity 1.1—Identify user/sponsor needs

The primary purpose of this activity is to develop a clear understanding of the problem to be addressed by the simulation environment. The needs statement may vary widely in terms of scope and degree of formalization. It should include, at a minimum, high-level descriptions of critical systems of interest, initial estimates of required fidelity and required behaviors for simulated entities, key events and environmental conditions that must be represented in the scenario, and output data requirements. In addition, the needs statement should indicate the resources that will be available to support the simulation environment (e.g., funding, personnel, tools, facilities) and any known constraints that may affect how the simulation environment is developed (e.g., required member applications, due dates, site requirements, security

requirements). In general, the needs statement should include as much detail and specific information as is possible at this early stage of the DSEEP.

An explicit and unambiguous statement of user/sponsor needs is critical to achieving clear communication of intent among the development/integration team responsible for the simulation environment. Failure to establish a common understanding of the required product can result in costly rework in later stages of the DSEEP.

4.1.1.1 Activity inputs

The potential inputs to this activity include the following. Neither this list of inputs, nor any subsequent lists of inputs, is meant to be completely exhaustive, nor are all mandatory for all simulation environments.

- Program goals
- Existing domain descriptions
- Information on available resources

4.1.1.2 Recommended tasks

The potential tasks for this activity include the following:

- Analyze the program goals to identify the specific purpose and objective(s) that motivate development and execution of a simulation environment.
- Identify available resources and known development and execution constraints.
- Document the information listed above in a needs statement.

Neither this list of tasks, nor any subsequent lists of tasks, is meant to be completely exhaustive, nor are all mandatory for all simulation environments.

4.1.1.3 Activity outcomes

The potential outcomes for this activity include the following. Neither this list of outcomes, nor any subsequent lists of outcomes, is meant to be completely exhaustive, nor are all mandatory for all simulation environments.

- Needs statement, including:
  - Purpose of the simulation environment
  - Identified needs (e.g., domain area/issue descriptions, high-level descriptions of critical systems of interest, initial estimates of required fidelity, and required behaviors for simulated entities)
  - Key events that must be represented in a scenario
  - Output data requirements
  - Resources that will be available to support the simulation environment (e.g., funding, personnel, tools, facilities)
  - Any known constraints that may affect how the simulation environment is developed and executed (e.g., due dates, security requirements)

4.1.2 Activity 1.2—Develop objectives

The purpose of this activity is to refine the needs statement into a more detailed set of specific objectives for the simulation environment. The objectives statement is intended as a foundation for generating explicit simulation requirements (i.e., translating high-level user/sponsor expectations into more concrete, measurable goals). This activity requires close collaboration between the user/sponsor of the simulation environment and the development/integration team to verify that the original needs statement is properly analyzed and interpreted correctly, and that the resulting objectives are consistent with the stated needs.

Early assessments of feasibility and risk should also be performed as part of this activity. In particular, certain objectives may not be achievable given practical constraints (such as cost, schedule, and availability of personnel or facilities) or even limitations on the state-of-the-art of needed technology. Early identification of such issues and consideration of these limitations and constraints in the *objectives statement* will set appropriate expectations for the development and execution effort.

4.1.2.1 Activity inputs

Potential inputs to this activity include the following:

— Needs statement

4.1.2.2 Recommended tasks

Potential tasks for this activity include the following:

— Analyze the needs statement.

— Define and document a prioritized set of objectives, consistent with the needs statement.

— Assess feasibility and risk, and incorporate into objectives statement.

— Develop metrics for each objective.

— Meet with the sponsor to review the objectives, and reconcile any differences.

— Begin planning discussions for the development and execution of the simulation environment.

4.1.2.3 Activity outcomes

Potential outcomes for this activity include the following:

— Objectives statement, including:

  — Potential solution approaches and rationale for the selection of a simulation environment as the best approach

  — A prioritized list of measurable objectives for the simulation environment

  — A high-level description of key simulation environment characteristics (repeatability, portability, time management approach, availability, etc.)

  — Domain context constraints or preferences, including object actions/relationships and natural environment representations (e.g., geographical regions covered, climate)

  — Identification of execution constraints to include functional (e.g., execution control mechanisms), technical (e.g., site, computational and network operations, health/performance

monitoring capabilities), economic (e.g., available funding), and political (e.g., organizational responsibilities)

- Identification of security needs and potential security risks, including probable security level and possible designated approval authority (or authorities, if a single individual is not possible)
- Identification of key evaluation measures to be applied to the simulation environment

4.1.3 Activity 1.3—Conduct initial planning

The purpose of this activity is to establish a preliminary simulation environment development and execution plan. The intent is to translate the objectives statement, along with the associated risk and feasibility assessments, into an initial plan with sufficient detail to effectively guide early design activities. The plan may effectively include multiple plans, and should cover such considerations as verification and validation (V&V), configuration management, and security. The plan should also address supporting tools for early DSEEP activities, based on factors such as availability, cost, applicability to the given application, ability to exchange data with other tools, and the personal preferences of the development/integration team.

The plan should also define a high-level schedule of key development and execution events, and provide additional scheduling detail for all pre-development (i.e., prior to Step 4) activities. Note that the initial plan will be updated and extended as appropriate in subsequent development phases as additional information is accumulated throughout the evolution of the distributed simulation design (see 4.3.3, Activity 3.3).

4.1.3.1 Activity inputs

Potential inputs to this activity include the following:

- Program goals
- Objectives statement

4.1.3.2 Recommended tasks

Potential tasks for this activity include the following:

- Define and document an initial simulation environment development and execution plan, V&V plan, test plan, configuration manatement plan, security plan, integration plan, and data management plan.
- Identify potential tools to support the initial plan.

4.1.3.3 Activity outcomes

Potential outcomes for this activity include the following:

- Initial planning documents, including:
  - Development and execution plan, including known schedule of activities, detailed task and identified milestones
  - V&V plan
  - Test plan

— Configuration management plan

— Security plan

— Integration plan

— Data management plan

— Management tools to support scenario development, conceptual analysis, requirements, VV&A, and configuration management

— Reusable products to facilitate reuse across DSEEP activities including supporting the development and update of designs, specifications, source code, documentation, test suites, manuals, and procedures, design, specification

— Simulation environment support tools to support data reduction, visualization, and simulation environment management

4.2 Step 2: Perform conceptual analysis

The purpose of this step of the DSEEP is to develop an appropriate representation of the real-world domain that applies to the defined problem space and to develop the appropriate scenario. It is also in this step that the objectives for the simulation environment are transformed into a set of highly specific requirements that will be used in during design, development, testing, execution, and evaluation.

Figure 4 illustrates the key activities in this step of the DSEEP. The subclauses that follow describe each of these activities in detail.

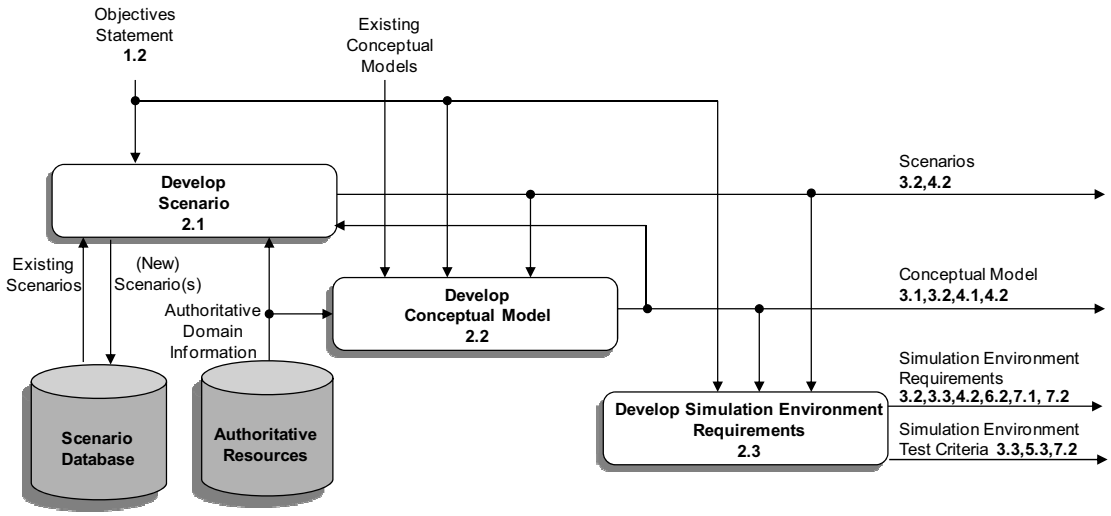

Figure 4—Perform conceptual analysis (Step 2)

4.2.1 Activity 2.1—Develop scenario

The purpose of this activity is to develop a functional specification of the scenario. Depending on the needs of the simulation environment, the scenario may actually include multiple scenarios, each consisting of one or more temporally ordered sets of events and behaviors (i.e., vignettes). The primary input to this activity is the domain constraints specified in the objectives statement (Step 1), although existing scenario databases may also provide a reusable starting point for scenario development. Any additional input is

provided by the conceptual model, which may be developed in parallel with the scenario. Where appropriate, authoritative sources for descriptions of major entities and their capabilities, behavior, and relationships should be identified prior to scenario construction. A scenario includes the types and numbers of major entities that must be represented within the simulation environment, a functional description of the capabilities, behavior, and relationships between these major entities over time, and a specification of relevant environmental conditions (such as urban terrain versus natural area, type of terrain, day/night, climate, etc.) that impact or are impacted by entities in the simulation environment. Initial conditions (e.g., geographical positions for physical objects), termination conditions, and specific geographic regions should also be provided. The product of this activity is a scenario or set of scenarios, which provides a bounding mechanism for conceptual modeling activities.

The presentation style used during scenario construction is at the discretion of the simulation environment development/integration team. Textual scenario descriptions, event-trace diagrams, and graphical illustrations of geographical positions for physical objects and communication paths all represent effective means of conveying scenario information. Software tools that support scenario development can generally be configured to produce these presentation forms. Reuse of existing scenario databases may also facilitate the scenario development activity.

4.2.1.1 Activity inputs

Potential inputs to this activity include the following:

- Objectives statement
- Existing scenarios
- Conceptual model(s)
- Authoritative domain specific documentation

4.2.1.2 Recommended tasks

Potential tasks for this activity include the following:

- Choose the appropriate tool/technique for development and documentation of the scenario(s).
- Using authoritative domain specific documentation, identify the entities, behaviors, and events that need to be represented in the scenario(s).
- Define one or more representative vignettes that, once executed, will produce the data necessary to achieve the objectives of the simulation environment.
- Define geographic areas of interest.
- Define environmental conditions of interest.
- Define initial conditions and termination conditions for the scenario(s).
- Select an appropriate scenario (or scenario set), or if new scenario information is to be developed, check with the stakeholder that the new scenario(s) will be acceptable.

4.2.1.3 Activity outcomes

Potential outcomes for this activity (in addition to potential planning document updates) include the following:

— Scenario(s), including:
  — Types and numbers of major entities/objects that must be represented within the simulation environment
  — Description of entity/object capabilities, behaviors, and relationships
  — Event timelines
  — Geographical region(s)
  — Natural environment condition(s)
  — Initial conditions
  — Termination conditions

4.2.2 Activity 2.2—Develop conceptual model

During this activity, the development/integration team produces a conceptual representation of the intended problem space based on their interpretation of user needs and sponsor objectives. The product resulting from this activity is known as a *conceptual model* (see Figure 4). The conceptual model provides an implementation-independent representation that serves as a vehicle for transforming objectives into functional and behavioral descriptions for system and software designers. The model also provides a crucial traceability link between the stated objectives and the eventual design implementation. This model can be used as the structural basis for many design and development activities (including scenario development) and can highlight correctable problems early in the development of the simulation environment when properly validated by the user/sponsor.

The early focus of conceptual model development is to identify relevant entities within the domain of interest, to identify static and dynamic relationships between entities, and to identify the behavioral and transformational (algorithmic) aspects of each entity. Static relationships can be expressed as ordinary associations or as more specific types of associations such as generalizations (“is-a” relationships) or aggregations (“part-whole” relationships). Dynamic relationships should include (if appropriate) the specification of temporally ordered sequences of entity interactions with associated trigger conditions. Entity characteristics (attributes) and interaction descriptors (parameters) may also be identified to the extent possible at this early stage of the process. While a conceptual model may be documented using differing notations, it is important that the conceptual model provides insight into the real-world domain and includes an explanatory listing of the assumptions and limitations to properly bound the model.

The conceptual model needs to be carefully evaluated before the next step (design simulation environment) is begun, including a review of key processes and events by the user/sponsor to confirm the adequacy of the domain representation. Revisions to the original objectives and conceptual model may be defined and implemented as a result of this feedback. As the conceptual model evolves, it is transformed from a general representation of the real-world domain to a more specific articulation of the capabilities of the simulation environment as constrained by the member applications of the simulation environment and available resources. The conceptual model will serve as a basis for many later development activities such as member application selection and simulation environment design, implementation, test, evaluation, and validation.

4.2.2.1 Activity inputs

Potential inputs to this activity include the following:

— Objectives statement
— Authoritative domain specific documentation

— Scenario(s)

— Existing conceptual models

4.2.2.2 Recommended tasks

Potential tasks for this activity include the following:

— Choose the technique and format for development and documentation of the conceptual model.

— Identify and describe all relevant entities within the domain of interest.

— Define static and dynamic relationships between identified entities.

— Identify events of interest within the domain, including temporal relationships.

— Capture applicable concept of operations in the conceptual model.

— Document the conceptual model and related decisions.

— Working with stakeholders, verify the contents of the conceptual model.

4.2.2.3 Activity outcomes

Potential outcomes for this activity (in addition to potential planning document updates) include the following:

— Conceptual model

4.2.3 Activity 2.3—Develop simulation environment requirements

As the conceptual model is developed, it will lead to the definition of a set of detailed requirements for the simulation environment. These requirements, based on the original objectives statement (Step 1), should be directly testable and should provide the implementation level guidance needed to design and develop the simulation environment. The requirements should consider the specific execution management needs of all users, such as execution control and monitoring mechanisms, data logging, etc. Such needs may also impact the scenario developed in Activity 2.1, see 4.2.1. The simulation environment requirements should also explicitly address the issue of fidelity, so that fidelity requirements can be considered during selection of simulation environment member applications. In addition, any programmatic or technical constraints on the simulation environment should be refined and described to the degree of detail necessary to guide implementation activities.

4.2.3.1 Activity inputs

Potential inputs to this activity include the following:

— Objectives statement

— Scenario(s)

— Conceptual model

4.2.3.2 Recommended tasks

Potential tasks for this activity include the following:

— Define required behaviors of identified entities and required characteristics of identified events.

— Define requirements for natural environment representation.

— Define requirements for live, virtual, and constructive simulations.

— Define human or hardware in-the-loop requirements.

— Define performance requirements for the simulation environment.

— Define evaluation requirements for the simulation environment.

— Define time management requirements (real time versus slower or faster than real time).

— Define host computer, networking, and other hardware requirements.

— Define supporting software requirements.

— Define security requirements for hardware, network, data, and software.

— Define output requirements, including requirements for data collection, raw execution data processing, and data analysis

— Define execution management requirements.

— Confirm that requirements are clear, unique, and testable.

— Develop simulation environment test criteria.

— Demonstrate traceability between requirements and program goals, simulation environment objectives, scenario(s), and conceptual model.

— Document all requirements for the simulation environment.

4.2.3.3 Activity outcomes

Potential outcomes for this activity (in addition to potential planning document updates) include the following:

— Simulation environment requirements

— Simulation environment test criteria

4.3 Step 3: Design simulation environment

The purpose of this step of the DSEEP is to produce the design of the simulation environment that will be implemented in Step 4. This involves identifying applications that will assume some defined role in the simulation environment (member applications) that are suitable for reuse, creating new member applications if required, allocating the required functionality to the member application representatives, and developing a detailed planning documents.

Figure 5 illustrates the key activities in this step of the DSEEP. The subclauses that follow describe each of these activities in detail.

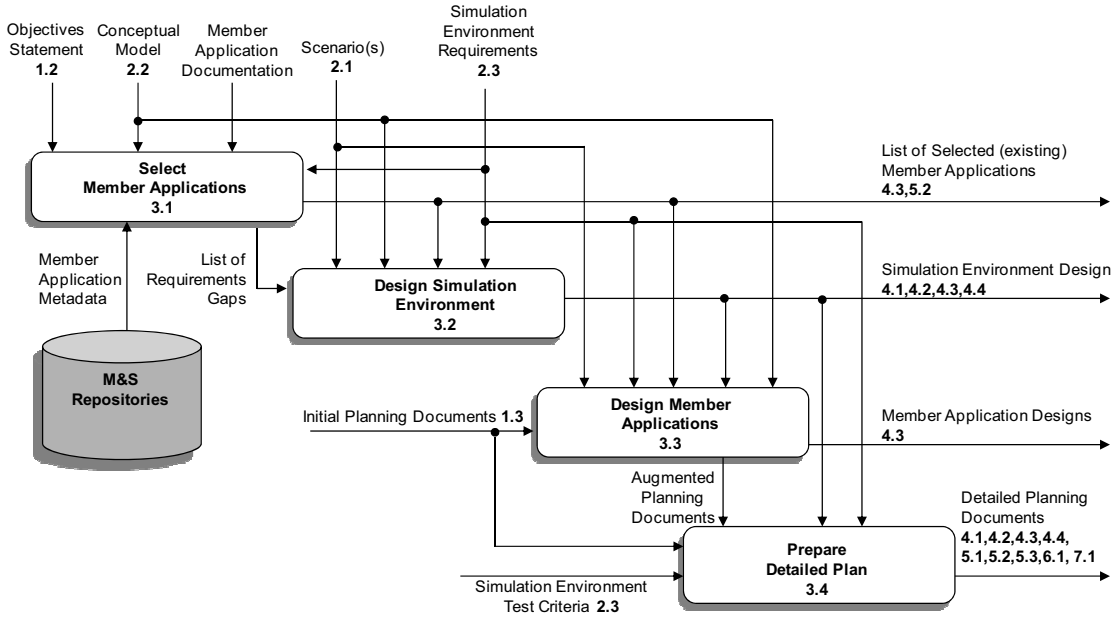

Figure 5—Design simulation environment (Step 3)

4.3.1 Activity 3.1—Select member applications

The purpose of this activity is to determine the suitability of individual simulation systems to become member applications of the simulation environment. This is normally driven by the perceived ability of potential member applications to represent entities and events according to the conceptual model. In some cases, these potential member applications may be simulation environments themselves, such as an aircraft simulation built from separate simulations of its subsystems. Metadata describing reusable and available member applications may be leveraged to discover candidate assets within existing M&S repositories. Managerial constraints (e.g., availability, security, facilities) and technical constraints (e.g., VV&A status, portability) may both influence the final selection of member applications.

In some simulation environments, the identity of at least some member applications will be known very early in the process. For instance, the sponsor may explicitly require the use of certain member applications in the simulation environment, or an existing simulation environment (with well-established member applications) may be reused and extended as necessary to address a new set of requirements. Although early member application selection may have certain advantages, it also introduces some immediate constraints on what the simulation environment will and will not be able to do. Since required capabilities are not always well understood at the initiation of the development activities, it is generally advisable to defer final decisions on simulation environment membership until this point in the overall process.

In some situations, it may be possible to satisfy the full set of requirements for the simulation environment with a single simulation. The use of a single simulation (with modifications as necessary) would eliminate many of the development and integration activities required for multi-member distributed simulation environments (as described in Steps 4 and 5). The developer/integrator should compare the time and effort required to perform the necessary modifications against the time and effort required to assimilate an established set of member applications into an integrated simulation environment. Other factors, such as reusability of the resulting software, should also be taken into account in deciding the proper design strategy.

Existing repositories should be searched for candidate member applications, keyed to critical entities and actions of interest. To support final member application selection decisions, additional information

resources (such as design and compliance documents) are generally necessary to fully understand internal simulation representations of required behaviors/activities and other practical use constraints. Finally, it may not be possible to make a firm decision between two competing member applications with the available data. If this situation arises, then both member applications may be taken to the design activity that follows to perform more detailed analysis and testing during the design activity.

4.3.1.1 Activity inputs

The potential inputs to this activity include the following:

- Objectives statement
- Conceptual model
- Simulation environment requirements (including required member applications)
- Member application documentation
- Member application metadata

4.3.1.2 Recommended tasks

The potential tasks for this activity include the following:

- Define criteria for member application selection.
- Determine if an existing, reusable simulation environment meets or partially satisfies the identified requirements.
- Search existing repositories for existing member applications and/or algorithms and code to be used by a member application.
- Identify candidate member applications (including predefined member applications).
- Analyze the ability of each candidate member application to represent required entities, events, and phenomena.
- Review overall purpose and objectives with respect to selected member applications and availability of resources.
- Document rationale (including assumptions) for selection of member applications.
- Document any requirements for which appropriate member applications cannot be identified.

4.3.1.3 Activity outcomes

The potential outcomes for this activity (in addition to potential planning document updates) include the following:

- List of selected (existing) member applications, including selection documentation such as criteria and rationale
- List of requirements gaps

4.3.2 Activity 3.2—Design simulation environment

Once all member applications have been identified, the next major activity is to prepare the simulation environment design and allocate the responsibility to represent the entities and actions in the conceptual

model to the member applications. This activity will allow for an assessment of whether the set of selected member applications provides the full set of required functionality. A by-product of the allocation of functionality to the member applications will be additional design information that can embellish the conceptual model.

A fundamental design choice for any distributed simulation environment is the underlying simulation architecture (e.g., HLA, DIS, TENA). In some cases, the requirements of the application will align with only one such architecture. In other cases, any of several different architectures could potentially satisfy the simulation environment requirements. In this latter case, several factors must be taken into account in making the final selection, such as the degree of training and experience of the development/integration team on each of the candidate architectures, adequacy of supporting data models, robustness and performance of the architecture middleware, and the availability and affordability of other required resources (e.g., tools, documentation). Note that the annexes in this document provide some basic, high-level guidance regarding the types of applications for which each identified simulation architecture is best suited.

In some large simulation environments, it is sometimes necessary to mix several simulation architectures. This poses special challenges to the simulation environment design, as sophisticated mechanisms are sometimes needed to reconcile disparities in the architecture interfaces. For instance, gateways or bridges to adjudicate between different on-the-wire protocols are generally a required element in the overall design, as well as mechanisms to address differences in SDEMs. Such mechanisms are normally formalized as part of the member application agreements, which are discussed in Step 4.

As agreements on assigned responsibilities are negotiated, various design trade-off investigations may be conducted as appropriate to support the development of the simulation environment design. Many of these investigations can be considered to be early execution planning and may include technical issues such as time management, execution management, infrastructure design, runtime performance, and potential implementation approaches.

The major inputs to this activity include the simulation environment requirements, the scenario, and the conceptual model (see Figure 5). In this activity, the conceptual model is used as a conduit to make sure that user domain-specific requirements are appropriately translated into the simulation environment design. High-level design strategies, including modeling approaches and/or tool selection, may be revisited and renegotiated at this time based on inputs from the member application representatives. When the simulation environment represents a modification or extension to a previous simulation environment, new member application representatives must be made cognizant of all relevant negotiated agreements and given the opportunity to revisit pertinent technical issues. For secure applications, efforts associated with maintaining a secure posture during the simulation execution can begin, including the designation of security responsibility. The initial security risk assessment and concept of operations may be refined at this time to clarify the security level and mode of operation.

4.3.2.1 Activity inputs

The potential inputs to this activity include the following:

- Conceptual model
- Scenario(s)
- Simulation environment requirements
- List of selected (existing) member applications
- List of requirements gaps

4.3.2.2 Recommended tasks

The potential tasks for this activity include the following:

- Analyze selected member applications and identify those that best provide required functionality and fidelity.
- Complete the allocation of functionality to all selected member applications and determine if modifications are necessary and/or if development of a new member application(s) is needed.
- Confirm that earlier decisions do not conflict with selected member applications.
- Evaluate alternative design options, and identify the simulation environment design that best addresses stated requirements.
- Develop design for simulation environment infrastructure, and select protocol standards and implementations.
- Evaluate the need for bridging technologies.
- Develop design of supporting databases.
- Estimate simulation environment performance, and determine if actions are necessary to meet performance requirements.
- Analyze, and if necessary, refine initial security risk assessment and concept of operations.
- Document the simulation environment design.
- Develop an experimental design as needed (e.g., for simulation environments with stochastic properties).

4.3.2.3 Activity outcomes

The potential outcomes for this activity (in addition to potential planning document updates) include the following:

- Simulation environment design, including:
  - Member application responsibilities
  - Simulation environment architecture (including supporting infrastructure design and standards to be supported)
  - Supporting tools (e.g., performance measurement equipment, network monitors)
  - Implied requirements for member application modifications and/or development of new member applications

4.3.3 Activity 3.3—Design member applications

In some circumstances, an existing set of member applications cannot fully address all defined requirements for the simulation environment. In these cases, it may be necessary to perform an appropriate set of design activities at the individual member application level. This may involve enhancements to one or more of the selected member applications, or could even involve designing an entirely new member application. The purpose of this activity is to transform the top-level design for the simulation environment into a set of detailed designs for the member applications. The scope of the design task will depend on the amount of previous design work that can be reused. New member applications will generally require a substantial amount of design effort whereas modifications to existing member applications will require less

effort. When existing member applications are being modified, it is necessary to document any changes to facilitate good configuration control.

4.3.3.1 Activity inputs

The potential inputs to this activity include the following:

- List of (existing) member applications
- Simulation environment requirements
- Simulation environment design
- Conceptual model
- Scenario
- Initial planning documents

4.3.3.2 Recommended tasks

The potential tasks for this activity include the following:

- Analyze simulation environment design.
- Design member applications.
- Formulate test strategy to verify design, and augment test plans as appropriate.
- Produce/update design documentation.

4.3.3.3 Activity outcomes

The potential outcomes for this activity include the following:

- Member application designs
- Augmented planning documents

4.3.4 Activity 3.4—Prepare detailed plan

Another major activity in Step 3, design simulation environment, is to develop a coordinated plan to guide the development, test, and execution of the simulation environment. This requires close collaboration among all member application representatives to obtain a common understanding of the various goals and requirements and also to identify (and agree to) appropriate methodologies and procedures based on recognized systems engineering principles. The initial planning documents prepared during development of the original objectives provide the basis for this activity (see Figure 5). The plan should include the specific tasks and milestones for each member application, along with proposed dates for completion of each task.

The plan may also identify the software tools that will be used to support the remaining life cycle of the simulation environment (e.g., CASE, configuration management, VV&A, testing). For new simulation environments, a plan to design and develop a network configuration may be required. These agreements, along with a detailed work plan, must be documented for later reference and possible reuse in future applications.

4.3.4.1 Activity inputs

The potential inputs for this activity include the following:

- Augmented planning documents
- Simulation environment requirements
- Simulation environment design
- Simulation environment test criteria

4.3.4.2 Recommended tasks

The potential tasks for this activity include the following:

- Refine and augment the initial simulation environment development and execution plan, including specific tasks and milestones for each member application.
- Revise V&V plan and test plan (based on simulation environment test criteria).
- Update the configuration management plan with plans and procedures for establishing and managing configuration baselines.
- Define security plan identifying needed simulation environment agreements and plans for securing these agreements.
- Develop integration plan and supporting methodology for simulation environment integration.
- Finalize the data management plan showing plans for data collection, management, and analysis.
- Complete selection of necessary management tools, reusable products, and simulation environment support tools, and develop plan for acquiring, installing, and utilizing these tools and resources.

4.3.4.3 Activity outcomes

The potential outcomes for this activity include the following:

- Detailed planning documents, including:
  - Development and execution plan, including schedule of activities, detailed tasks, and milestones
  - V&V plan
  - Test plan
  - Configuration management plan
  - Security plan
  - Integration plan
  - Data management plan
  - Management tools
  - Reusable products
  - Simulation environment support tools

4.4 Step 4: Develop simulation environment

The purpose of this step of the DSEEP is to define the information that will be exchanged at runtime during the execution of the simulation environment, modify member applications if necessary, and prepare the simulation environment for integration and test (database development, security procedure implementation, etc.).

Figure 6 illustrates the key activities in this phase of the DSEEP. The subclauses that follow describe each of these activities in detail.

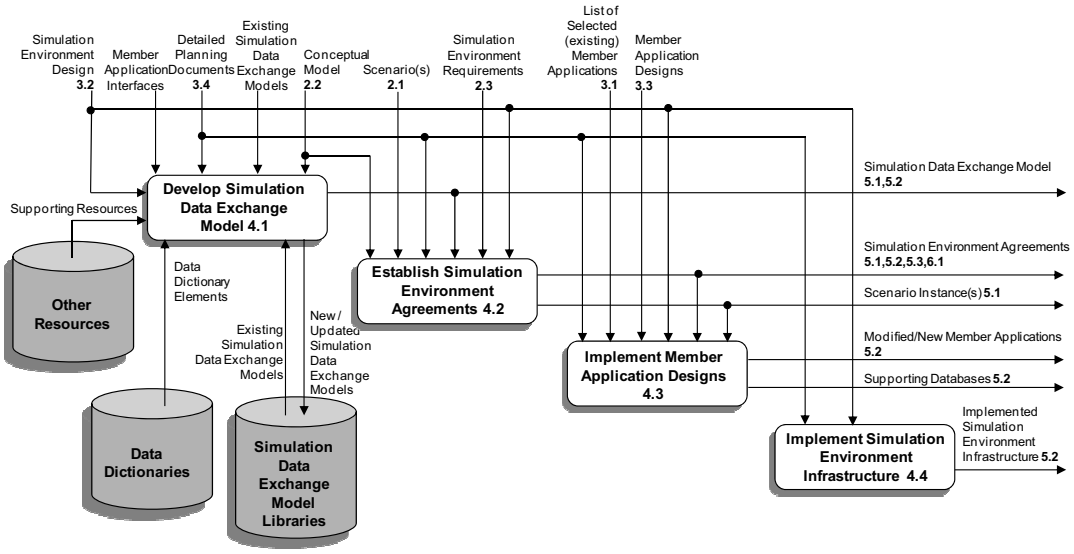

Figure 6—Develop simulation environment (Step 4)

4.4.1 Activity 4.1—Develop simulation data exchange model

In order for the simulation environment to operate properly, there must be some means for member applications to interact. At a minimum, this implies the need for runtime data exchange, although it could also involve remote method invocations or other such means of direct interaction among cooperating object representations. Clearly, there must be agreements among the member applications as to how these interactions will take place, defined in terms of software artifacts like class relationships and data structures. Collectively, the set of agreements that govern how this interaction takes place is referred to as the *simulation data exchange model (SDEM)*.

Depending on the nature of the application, the SDEM may take several forms. Some simulation applications are strictly object-oriented, where both static and dynamic views of the simulation system are defined in terms of class structures, class attributes, and class operations (i.e., methods) within the SDEM. Other simulation applications maintain this same object-based paradigm, but use object representations as a way to share state information among different member applications about entities and events that are being modeled internal to the member applications themselves. In this case, the SDEM is quite similar to a data model, including a defined set of format, syntax, and encoding rules. Still other simulation applications may not use an object-based structure at all in their SDEM. Rather, the focus is on the runtime data structures themselves and the conditions that cause the information to be exchanged. In general, different applications will have different requirements for the depth and nature of interaction among member applications, and while these varying requirements will drive the type and content of the SDEM, it is necessary for the SDEM to exist in order to formalize the “contract” among member applications necessary to achieve coordinated and coherent interoperation within the simulation environment.

There are many possible approaches to SDEM development. Certainly, reuse of an existing SDEM is the most expeditious approach, if one can be found within a repository that meets all member application interaction requirements. If an exact match for the needs of the current application cannot be found, then identifying an SDEM that meets some of these needs and modifying/tailoring the SDEM to meet the full set of requirements would generally be preferable to starting from a "clean sheet." Some communities maintain reference SDEMs for their users to facilitate this type of reuse (e.g., the real-time platform reference federation object model (RPR FOM); see SISO-STD-0001.1 [B2]). Still other approaches involve assembling an SDEM from small, reusable SDEM components (e.g., base object models (BOM), see SISO-STD-003 [B3]) and merging SDEM elements from the interfaces of member applications [e.g., HLA simulation object models (SOM)]. In general, the simulation environment development/integration team should employ whatever approach makes most sense from a cost, efficiency, and quality perspective.

4.4.1.1 Activity inputs

The potential inputs to this activity include the following:

- Simulation environment design
- Member application interfaces
- Detailed planning documents
- Data dictionary elements
- Existing SDEMs
- Supporting resources (e.g., SDEM development tools, SDEM libraries, dictionaries)
- Conceptual model

4.4.1.2 Recommended tasks

The potential tasks for this activity include the following:

- Choose an SDEM development approach.
- Identify appropriate SDEMs or SDEM subsets for reuse.
- Review applicable data dictionaries to identify relevant SDEM elements.
- Develop and document the SDEM using an appropriate tool.
- Verify that the SDEM supports the conceptual model.

4.4.1.3 Activity outcomes

The potential outcomes for this activity (in addition to potential planning document updates) include the following:

- Simulation data exchange model

4.4.2 Activity 4.2—Establish simulation environment agreements

Although the SDEM represents an agreement among member application representatives as to how runtime interaction will take place, there are other operating agreements that must be reached that are not documented in the SDEM. Such agreements are necessary to establish a fully consistent, interoperable, simulation environment. While the actual process of establishing agreements among all participants in the

development effort begins early in the DSEEP and is embodied in each of its activities, this may not result in a complete set of formally documented agreements. It is at this point in the overall process that the development/integration team needs to explicitly consider what additional agreements are required and how they should be documented.

There are many different types of such agreements. For instance, the development/integration team uses the conceptual model to gain an understanding of the necessary agreements and anticipated behavior of all entities within the simulation environment. While behaviors that involve multiple objects will be documented to some extent in the SDEM, other means may be required to fully address these issues. Additional requirements for software modifications to selected member applications may be identified as a result of these discussions; such requirements must be addressed prior to integration activities. Also, agreements must be reached as to the databases and algorithms that must be common (or at least consistent) across the simulation environment to achieve valid interactions among all member applications. For instance, a consistent view of the features and phenomena represented across the entire simulated environment is critical in order for objects owned by different member applications to interact and behave in a realistic fashion. In addition, certain operational issues must be addressed and resolved among all member application representatives. For instance, agreements on initialization procedures, synchronization points, save/restore policies, and security procedures are all desirable to facilitate proper operation of the simulation environment.

Once all authoritative data sources that will be used in support of the simulation environment have been identified, the actual data stores are used to transition the functional description of the scenario (developed in Step 2; see Figure 4) to an executable scenario instance (or set of instances). The product of this activity permits testing to be conducted directly within the context of the domain of interest and also drives the execution of the simulation environment later in the DSEEP.

Finally, the development/integration team must recognize that certain agreements may require the activation of other processes external to the DSEEP. For instance, utilization and/or modification of certain member applications may require contractual actions between the user/sponsor and the development/integration team. Even where contractual actions may not be required, formal memoranda of agreement may be required between member application representatives. Additionally, simulation environments requiring the processing of classified data will generally require the establishment of a security agreement between the appropriate security authorities. Each of these external processes has the potential to negatively impact the development and execution of a simulation environment within resource and schedule constraints and should be factored into project plans as early as possible.

4.4.2.1 Activity inputs

The potential inputs to this activity include the following:

- Scenario(s)
- Conceptual model
- Simulation environment design
- Detailed planning documents
- Simulation environment requirements
- Simulation data exchange model

4.4.2.2 Recommended tasks

The potential tasks for this activity include the following:

— Decide the behavior of all objects within the simulation environment.

— Identify the necessary software modifications to selected member applications that were not previously identified.

— Decide which databases and algorithms must be common or consistent.

— Identify authoritative data sources for both member application and simulation environment databases.

— Decide how time should be managed in the simulation environment.

— Establish synchronization points for the simulation environment and the procedures for initialization.

— Decide strategy for how the simulation environment shall be saved and restored.

— Decide how data is to be distributed across the simulation environment.

— Transform the functional scenario description to an executable scenario [scenario instance(s)].

— Review security agreements, and establish security procedures.

4.4.2.3 Activity outcomes

The potential outcomes for this activity (in addition to potential planning document updates) include the following:

— Simulation environment agreements, including:
  — Established security procedures
  — Time management agreements
  — Data management and distribution agreements
  — Defined synchronization points and initialization procedures
  — Save/restore strategy
  — Agreements on supporting databases and algorithms
  — Agreements on authoritative data sources
  — Agreements on publication and subscription responsibilities
— Scenario instance(s)

4.4.3 Activity 4.3—Implement member application designs

The purpose of this activity is to implement whatever modifications are necessary to the member applications so that they can represent assigned objects and associated behaviors as described in the conceptual model (Step 2), produce and exchange data with other member applications as defined by the SDEM, and abide by the established simulation environment agreements. This may require internal modifications to the member application to support assigned domain elements, or it may require modifications or extensions to the member application's external interface to support new data structures or services that were not supported in the past. In some cases, it may even be necessary to develop a whole new interface for the member application, depending on the content of the SDEM and simulation environment agreements. In this situation, the member application representative must consider both the resource (e.g., time, cost) constraints of the immediate application as well as longer-term reuse issues in deciding the best overall strategy for completing the interface. In situations where entirely new member applications are needed, the implementation of the member application design must take place at this time.

4.4.3.1 Activity inputs

The potential inputs to this activity include the following:

- Detailed planning documents
- List of selected (existing) member applications
- Member application designs
- Simulation environment design
- Simulation environment agreements
- Scenario instance(s)

4.4.3.2 Recommended tasks

The potential tasks for this activity include the following:

- Implement member application modifications to support allocated functionality.
- Implement modifications of, or extensions to, the interfaces of all member applications.
- Develop a new interface for those member applications for which it is required.
- Implement design of new member applications as required.
- Implement and populate supporting databases and scenario instance(s).

4.4.3.3 Activity outcomes

The potential outcomes for this activity (in addition to potential planning document updates) include the following:

- Modified and/or new member applications
- Supporting databases

4.4.4 Activity 4.4—Implement simulation environment infrastructure

The purpose of this activity is to implement, configure, and initialize the infrastructure necessary to support the simulation environment and verify that it can support the execution and intercommunication of all member applications. This involves the implementation of the network design [e.g., wide area networks (WANs), local area networks (LANs)]; the initialization and configuration of the network elements (e.g., routers, bridges); and the installation and configuration of supporting software on all computer systems. This also involves whatever facility preparation is necessary to support integration and test activities.

4.4.4.1 Activity inputs

The potential inputs to this activity include the following:

- Simulation environment design
- Detailed planning documents

4.4.4.2 Recommended tasks

The potential tasks for this activity include the following:

- Prepare integration/test facility, including:
  - Confirm basic facility services (air conditioning, electric power, etc.) are functional and available.
  - Confirm availability of required hardware/software in integration/test facility.
  - Perform required system administration functions (establish user accounts, establish procedures for file backups, etc.).
- Implement infrastructure design, including:
  - Install and configure required hardware elements.
  - Install and configure middleware (e.g., a runtime infrastructure (RTI) component for HLA) and/or other supporting software.
  - Test infrastructure to verify proper operation.
- Confirm that the infrastructure adheres to the security plan.

4.4.4.3 Activity outcomes

The potential outcomes for this activity (in addition to potential planning document updates) include the following:

- Implemented simulation environment infrastructure

4.5 Step 5: Integrate and test simulation environment

The purpose of this step of the DSEEP is to plan the execution of the simulation, establish all required interconnectivity between member applications, and test the simulation environment prior to execution.

Figure 7 illustrates the key activities in this step of the DSEEP. The subclauses that follow describe each of these activities.

4.5.1 Activity 5.1—Plan execution

The main purpose of this activity is to fully describe the execution environment and develop an execution plan. For instance, performance requirements for individual member applications and for the larger simulation environment along with salient characteristics of host computers, operating systems, and networks that will be used in the simulation environment should all be documented at this time. The completed set of information, taken together with the SDEM and simulation environment agreements, provides the necessary foundation to transition into the integration and testing phase of development.

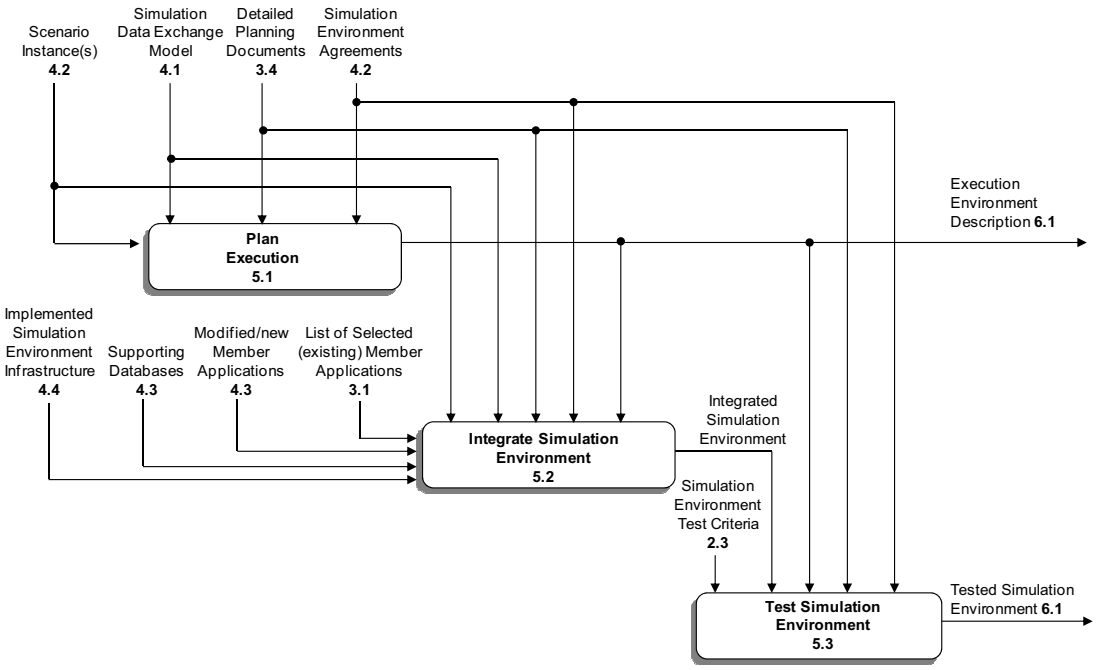

Figure 7—Integrate and test simulation environment (Step 5)

Additional activities in this step include the incorporation of any necessary refinements to test and VV&A plans and (for secure environments) the development of a security test and evaluation plan. This latter activity requires reviewing and verifying the security work accomplished thus far in the simulation environment development and finalizing the technical details of security design, such as information downgrading rules, formalized practices, etc. This plan represents an important element of the necessary documentation set for the simulation environment.

Operational planning is also a key aspect of this activity. This planning should address which personnel will be operating the member applications (operational personnel) or supporting the simulation execution (support personnel) in other ways (e.g., monitoring, data logging). It should detail the schedule for both the execution runs and the necessary preparation prior to each run. Training and rehearsal for support and operational personnel should be addressed as necessary. Specific procedures for starting, conducting, and terminating each execution run should be documented.

4.5.1.1 Activity inputs

The potential inputs to this activity include the following:

— Simulation data exchange model

— Scenario instance(s)

— Simulation environment agreements

— Detailed planning documents

4.5.1.2 Recommended tasks

The potential tasks for this activity include the following:

— Refine/augment planning documents in the areas of execution, VV&A, integration, test, data collection, and security.

— Assign member applications to appropriate infrastructure elements.

— Identify risks, and add risk mitigation activities (as required) to the simulation environment development and execution plan component.

— Define all support and operational personnel training needs for simulation execution and proposed methods of resolution.

— Document all information relevant to the execution.

— Include appropriate "pass/fail" criteria in the simulation environment development and execution plan component.

4.5.1.3 Activity outcomes

The potential outcomes for this activity (in addition to potential planning document updates) include the following:

— Execution environment description (including hardware/software configurations)

4.5.2 Activity 5.2—Integrate simulation environment

The purpose of this activity is to bring all of the member applications into a unifying operating environment. This requires that all hardware and software assets are properly installed and interconnected in a configuration that can support the SDEM and simulation environment agreements. The simulation environment development and execution plan, which is a component of the detailed planning documents, specifies the integration methodology used in this activity, and the scenario instance provides the necessary context for integration activities.

Integration activities are normally performed in close coordination with testing activities. Iterative "test-fix-test" approaches are used quite extensively in practical applications and have been shown to be quite effective.

4.5.2.1 Activity Inputs

The potential inputs to this activity include the following:

— Detailed planning documents, which includes the simulation environment development and execution plan

— Execution environment description

— Simulation environment agreements

— Scenario instance

— Simulation data exchange model

— Member applications, including:

  — List of selected (existing) member applications, and

  — Modified and/or newly developed member applications

— Implemented simulation environment infrastructure

— Supporting databases

4.5.2.2 Recommended tasks

The potential tasks for this activity include the following:

— Confirm that all member application software is properly installed and interconnected.

— Establish method for managing known software problems and "workarounds."

— Perform incremental integration according to the integration plan.

— Provide planned support and operational personnel training on the fully integrated simulation environment.

4.5.2.3 Activity outcomes

The potential outcomes for this activity (in addition to potential planning document updates) include the following:

— Integrated simulation environment

4.5.3 Activity 5.3—Test simulation environment

The purpose of this activity is to test that all of the member applications can interoperate to the degree required to achieve core objectives. Three levels of testing are defined for simulation applications as follows:

Member application testing: In this activity, each member application is tested to confirm that the member application software correctly implements its role in the simulation environment as documented in the SDEM, execution environment description, and any other operating agreements.

Integration testing: In this activity, the simulation environment is tested as an integrated whole to verify a basic level of interoperability. This testing primarily includes observing the ability of the member applications to interact correctly with the supporting infrastructure and to communicate with other member applications as described by the SDEM.

Interoperability testing: In this activity, the ability of the simulation environment to interoperate to the degree necessary to achieve identified objectives is tested. This includes observing the ability of member applications to interact according to the defined scenario and to the level of fidelity required for the application. This activity also includes security certification testing if required for the application. The results from interoperability testing may contribute to V&V of the simulation environment as required.

Procedures for conducting interoperability testing must be agreed upon by all member application representatives and documented appropriately. Data collection plans should be exercised during the testing phase to confirm that the data needed to support the overall objectives is being accurately collected and stored.

The desired output from this activity is an integrated, tested, validated, and if required, accredited simulation environment that indicates that execution of the simulation environment can commence. If early testing and validation uncover obstacles to successful integration and accreditation, the development/integration team must take corrective actions. In many cases, these corrective actions simply require a relatively minor software fix (or series of fixes) or minor adjustment to the SDEM. However, testing may also uncover more serious software, interoperability, or validity problems. In these cases, options may need to be identified, with their associated cost and schedule estimates (including security and

VV&A implications), and should be discussed with the user/sponsor of the simulation environment before corrective action is taken.

4.5.3.1 Activity inputs

The potential inputs to this activity include the following:

- Detailed planning documents, which includes the simulation environment development and execution plan
- Simulation environment agreements
- Execution environment description
- Integrated simulation environment
- Simulation environment test criteria

4.5.3.2 Recommended tasks

The potential tasks for this activity include the following:

- Rehearse the execution of the tested simulation environment to identify any unforeseen problems with the integrated environment or its use by operational personnel.
- Perform member application-level testing.
- Perform connectivity and interoperability testing across the full simulation environment.
- Analyze testing results (i.e., compare against simulation environment test criteria).
- Review test results with user/sponsor.

4.5.3.3 Activity outcomes

The potential outcomes for this activity (in addition to potential planning document updates) include the following:

- Tested (and if necessary, accredited) simulation environment, including
  - Member application test data
  - Tested member applications
  - Simulation environment test data
  - Corrective actions

4.6 Step 6: Execute simulation

The purpose of this step of the DSEEP is to execute the integrated set of member applications (i.e., the "simulation") and to preprocess the resulting output data.

Figure 8 illustrates the key activities in this step of the DSEEP. The subclauses that follow describe each of these activities.

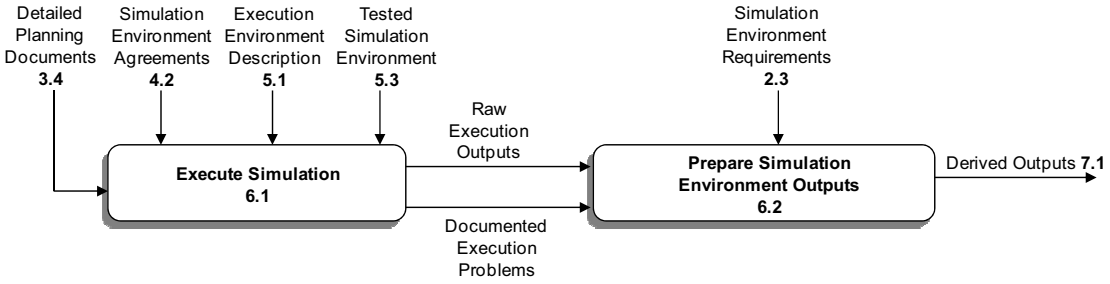

Figure 8—Execute simulation (Step 6)

4.6.1 Activity 6.1—Execute simulation

The purpose of this activity is to exercise all member applications of the simulation environment in a coordinated fashion over time to generate required outputs, and thus achieve stated objectives. The simulation environment must have been tested successfully before this activity can begin.

Execution management and data collection are critical to a successful simulation execution. Execution management involves controlling and monitoring the execution via specialized software tools (as appropriate). Execution can be monitored at the hardware level (e.g., CPU usage, network load), and/or software operations can be monitored for individual member applications or across the full simulation environment. During execution, key simulation environment test criteria should be monitored to provide an immediate evaluation of the successful execution of the simulation.

Data collection is focused on assembling the desired set of outputs and on collecting whatever additional supporting data is required to assess the validity of the execution. In some cases, data is also collected to support replays of the execution (i.e., “playbacks”). Essential runtime data may be collected via databases in the member applications themselves, or can be collected via specialized data collection tools directly interfaced to the simulation environment infrastructure. The particular strategy for data collection in any particular simulation environment is entirely at the discretion of the development/integration team, and should have been documented in the simulation environment requirements, the detailed planning documents, and in the simulation environment agreements.

When security restrictions apply, strict attention must be given to maintaining the security posture of the simulation environment during execution. A clear concept of operations, properly applied security measures, and strict configuration management will all facilitate this process. It is important to remember that authorization to operate is usually granted for a specific configuration of member applications. Any change to the member applications or composition of the simulation environment will certainly require a security review and may require some or all of the security certification tests to be redone.

4.6.1.1 Activity inputs

The potential inputs to this activity include the following:

— Tested simulation environment

— Detailed planning documents

— Simulation environment agreements

— Execution environment description

4.6.1.2 Recommended tasks

The potential tasks for this activity include the following:

- Perform identified executions and collect data.
- Manage the execution in accordance with the detailed planning documents.
- Document detected problems during execution.
- Confirm secure operation in accordance with certification and accreditation decisions and requirements.

4.6.1.3 Activity outcomes

The potential outcomes for this activity (in addition to potential planning document updates) include the following:

- Raw execution outputs (data)
- Documented execution problems

4.6.2 Activity 6.2—Prepare simulation environment outputs

The purpose of this activity is to preprocess the output collected during the execution, in accordance with the specified requirements, prior to the formal analysis of the data in Step 7. This may involve the use of data reduction techniques to reduce the quantity of data to be analyzed and to transform the data to a particular format. Where data has been acquired from many sources, data fusion techniques may have to be employed. The data should be reviewed and appropriate action taken where missing or erroneous data is suspected. This may require further executions to be conducted.

4.6.2.1 Activity inputs

The potential inputs to this activity include the following:

- Requirements for raw data preprocessing
- Raw execution outputs (data)
- Documented execution problems

4.6.2.2 Recommended tasks

The potential tasks for this activity include the following:

- Merge data from multiple sources
- Reduce/transform raw data
- Review data for completeness and possible errors

4.6.2.3 Activity outcomes

The potential outcomes for this activity (in addition to potential planning document updates) include the following:

— Derived outputs

4.7 Step 7: Analyze data and evaluate results

The purpose of this step of the DSEEP is to analyze and evaluate the data acquired during the execution of the simulation environment (Step 6), and to report the results back to the user/sponsor. This evaluation is necessary to confirm that the simulation environment fully satisfies the requirements of the user/sponsor. The results are fed back to the user/sponsor so that they can decide if the original objectives have been met or if further work is required. In the latter case, it will be necessary to repeat some of the DSEEP steps again with modifications to the appropriate products.

Figure 9 illustrates the key activities in this step of the DSEEP. The subclauses that follow describe each of these activities.

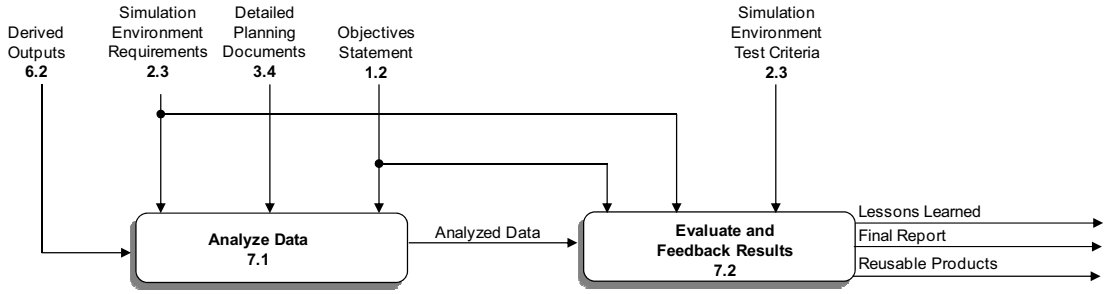

Figure 9—Analyze data and evaluate results (Step 7)

4.7.1 Activity 7.1—Analyze data

The main purpose of this activity is to analyze the execution data from Step 6. This data may be supplied using a range of different media (e.g., digital, video, audio), and appropriate tools and methods will be required for analyzing the data. These may be commercial or government off-the-shelf (COTS/GOTS) tools or specialized tools developed for a specific simulation environment. The analysis methods used will be specific to a particular simulation environment and can vary between simple observations (e.g., determining how many targets have been hit) to the use of complex algorithms (e.g., regression analysis or data mining). In addition to data analysis tasks, this activity also includes defining appropriate formats for presenting results to the user/sponsor.

4.7.1.1 Activity inputs

The potential inputs to this activity include the following:

— Simulation environment requirements
— Derived outputs
— Detailed planning documents
— Objectives statement

4.7.1.2 Recommended tasks

The potential tasks for this activity include the following:

- Apply analysis methods and tools to data
- Define appropriate presentation formats
- Prepare data in chosen formats

4.7.1.3 Activity outcomes

The potential outcomes for this activity (in addition to potential planning document updates) include the following:

- Analyzed data

4.7.2 Activity 7.2—Evaluate and feedback results

There are two main evaluation tasks. In the first task, the derived results from the previous activity are evaluated to determine if all objectives have been met. This requires a retracing of execution results to the measurable set of requirements originally generated during conceptual analysis (Step 2) and refined in subsequent DSEEP steps. This step also includes evaluating the results against the test and execution "pass/fail" criteria for the simulation environment. In the vast majority of cases, any impediments to fully satisfying the requirements have already been identified and resolved during the earlier development and integration phases. Thus, for well-designed simulation environments, this task is merely a final check. Having completed this first evaluation process, the conclusions should be communicated to the user/sponsor. In those rare cases in which certain objectives have not been fully met at this late stage of the overall process, then, with the user/sponsor's approval as there may be cost and time implications, corrective actions should be identified and implemented. This may necessitate revisiting previous steps of the DSEEP and regenerating results.

The second evaluation task in this activity is to assess all products generated in terms of their reuse potential within the domain or broader user community. Those products identified as having such reuse potential should be stored in an appropriate archive. Examples of potentially reusable products include the scenario and the conceptual model, although there may be several others. In fact, it may be advantageous in some instances to capture the full set of products required to reproduce the execution of the simulation environment. Determination of which products have potential for reuse in future applications is at the discretion of the development/integration team.

4.7.2.1 Activity inputs

The potential inputs to this activity include the following:

- Analyzed data
- Objectives statement
- Simulation environment requirements
- Simulation environment test criteria

4.7.2.2 Recommended tasks

The potential tasks for this activity include the following:

— Determine if all objectives for the simulation environment have been met.

— Provide feedback and conclusions to user/sponsor.

— With agreement from user/sponsor, consider appropriate corrective actions if deficiencies are found.

— Archive all reusable products.

4.7.2.3 Activity outcomes

The potential outcomes for this activity (in addition to potential planning document updates) include the following:

— Lessons learned

— Final report

— Reusable products

5. Conclusion

This document has provided a view of the DSEEP, which represents the best practices available to the distributed simulation user community. The DSEEP is an easily tailored process and is offered as guidance to all users, developers, and managers of distributed simulation environments. As additional experience is accrued in building such applications, the DSEEP will leverage this knowledge and evolve accordingly.

The DSEEP has been designed to serve as the generalized framework from which alternative; more detailed views can be specified in order to better serve the specialized needs of specific communities. Such views provide more detailed “hands-on” guidance to users of this process from the perspective of a particular domain (e.g., analysis, training), a particular discipline (e.g., VV&A, security), or a particular implementation strategy (e.g., HLA, DIS, TENA). Some selected examples of this latter case are provided as annexes to this document. In general, participants in distributed simulation activities are encouraged to perform these types of adaptations whenever appropriate.

Annex A

(normative)

High level architecture process overlay to the DSEEP

A.1 Introduction

High Level Architecture (HLA) is a technical architecture developed to facilitate the reuse and interoperation of simulation systems and assets (e.g., federation managers, data collectors, passive viewers, real-world ("live") systems, and other utilities). It was developed in response to the clear recognition of the value of modeling and simulation in supporting a wide variety of applications, as well as the need to manage M&S assets as cost-effective tools. It is intended as a general-purpose architecture, in that a very wide range of standardized services and capabilities are provided that collectively meets the needs of a very broad spectrum of potential users. For instance, HLA provides time management services for those simulation environments that require either faster or slower than real-time operation, advanced filtering capabilities for very large-scale applications (with scalability concerns), and services to manage the operation of the simulation environment. While most applications only employ a defined subset of the full range of HLA services, it is believed that the union of all HLA services is sufficient to address the needs of any class of user within the larger distributed simulation community.

A complete description of the HLA is provided in two different sources. The U.S. Department of Defense (DoD) sponsored the original development of the HLA, which produced a series of stepwise revisions culminating with the HLA Version 1.3 specifications including the Interface Specification, the Object Model Template (OMT) Specification, and the Rules (see HLA Interface Specification [B8], HLA Object Model Template [B9], HLA Rules [B10],). These specifications are still in active use within various U.S. DoD user communities today. Later, an effort to develop an industry standard for the HLA was initiated in collaboration with the IEEE, resulting in the current IEEE 1516™ series of HLA documents: IEEE Std 1516™ [B5], IEEE Std 1516.1™ [B6], IEEE Std 1516.2™ [B7]. Both the HLA v1.3 and IEEE 1516 specifications are presently available to users, as are commercial and government tools/utilities to facilitate the development of HLA applications.

A.2 Terminology mappings and definitions (specific architecture/methodology to DSEEP)

A.2.1 Terminology mappings

Simulation environment:	Federation
Member application:	Federate
Simulation data exchange model:	Federation object model (FOM) or simulation object model (SOM)

A.2.2 Definitions

federation: A named set of federate applications and a common federation object model (FOM) that are used as a whole to achieve some specific objective.

federate: A member of a federation; a single application that may be or is currently coupled with other software applications under a *federation execution data* (FED) or *federation object model document data* (FDD) and a runtime infrastructure (RTI).

federation object model (FOM): A specification defining the information exchanged at runtime to achieve a given set of federation objectives. This includes object classes, object class attributes, interaction classes, interaction parameters, and other relevant information.

simulation object model (SOM): A specification of the types of information that an individual federate could provide to HLA federations as well as the information that an individual federate can receive from other federates in HLA federations.

A.3 Global mapping (specific architecture/methodology to DSEEP)

With appropriate term translations, the specific process used for developing and executing an HLA federation[4] (see IEEE Std 1516.3™-2003) is essentially the same as the DSEEP. That is, the HLA development process has the exact same number of steps (seven), and each step can be decomposed down into the same basic set of activities. Besides the obvious differences in terminology, the main differences are with respect to how the generic activities and lower-level tasks described in the DSEEP are implemented in an HLA context. The following subclauses identify the major steps and activities required to build and execute an HLA federation, including a summary of the differences with the more generic DSEEP activity descriptions.

A.4 Detailed mappings (specific architecture/methodology to DSEEP)

A.4.1 Step 1: Define federation objectives

A.4.1.1 Activity 1.1—Identify user/sponsor needs

This activity is identical to that described in the DSEEP.

A.4.1.2 Activity 1.2—Develop federation objectives

This activity is essentially identical to that described in the DSEEP. However, the *federation objectives statement* may contain information unique to HLA applications, such as rationale for the selection of HLA as the preferred architecture/methodology and possible sponsor recommendations and/or constraints on the choice of RTI and other HLA tools.

A.4.2 Step 2: Perform conceptual analysis

A.4.2.1 Activity 2.1—Develop federation scenario

Since the scenario description is independent of the simulation protocol, this activity is identical with the DSEEP.

[4] Formally known as the HLA Federation Development and Execution Process (FEDEP).

A.4.2.2 Activity 2.2—Develop federation conceptual model

Like the scenario description, the *federation conceptual model* is also independent of the simulation protocol, and thus this activity is also identical with the DSEEP.

A.4.2.3 Activity 2.3—Develop federation requirements

While the necessary tasks to produce a set of testable requirements are essentially the same as other architectures and methodologies, there may be some types of requirements that are unique to HLA. For instance, requirements for runtime services in such areas as time management and data distribution management may require the use of HLA, and in fact may be the primary rationale for the selection of HLA as the supporting architecture.

A.4.3 Step 3: Design federation

A.4.3.1 Activity 3.1—Select federates

The basic activity to select the participants in the federation is essentially the same as the DSEEP. However, while the DSEEP identifies a generic set of M&S repositories as the principal source of candidate participants, the HLA development process identifies existing HLA object model libraries as the primary means of federate selection. That is, existing FOM are used to identify federates that have previously participated in similar simulation environments, while existing SOM are used to evaluate the ability of individual federates to support key entities and interactions in current federation applications. While other sources can also be used to identify candidate federates, existing FOM/SOM libraries are considered the main source.

There are also a number of HLA-specific factors that may affect federate selection. For instance, if time management services will be used in the federation, the ability of potential federates to support HLA time management services will be an important consideration. In general, the ability of a potential federate to support required federation services will be a critical factor in selecting that federate for the current federation application.

A.4.3.2 Activity 3.2—Prepare federation design

While the need to prepare the design of the simulation environment is obviously necessary for any architecture/methodology, the decision to employ HLA has certain implications for the design. For instance (per the HLA Rules), all exchange of FOM data must be through the RTI, and all object instance representations must be in the federates (not the RTI). In general, the federation design must describe how the various HLA services (as defined in the HLA Interface Specification) will be used to enable the operation of federation and achieve the core federation goals. The generalized considerations of security, modeling approaches, and tool selection as identified in the DSEEP are also applicable to HLA federations.

A.4.3.3 Activity 3.3—Prepare plan

The general process of planning for a simulation environment development is essentially the same for any supporting architecture or methodology. Thus, this activity can generally be considered to be identical to the corresponding activity in the DSEEP. However, the resulting plan must capture all HLA-specific considerations, such as RTI and *object model development tool* (OMDT) selection and the strategy for implementation of HLA services.

A.4.4 Step 4: Develop federation

A.4.4.1 Activity 4.1—Develop SDEM

In the DSEEP, the process of formally defining the SDEM is described in very general terms in order to provide equal support for the many similar types of products used within the simulation community. In the HLA, the SDEM is synonymous with the FOM. A FOM is described as "a specification defining the information exchanged at runtime to achieve a given set of federation objectives. This includes object classes, object class attributes, interaction classes, interaction parameters, and other relevant information." The HLA *object model template* (OMT), one of the three HLA specifications, prescribes the required format and syntax for all HLA object models, although the actual content of the runtime data exchange is application-dependent. Thus, to accomplish this activity for HLA applications, it is necessary to define the runtime data exchange in terms of the set of tables defined within the OMT.

There are several HLA-specific resources that are useful for building HLA FOMs. HLA object model libraries (if they exist) can provide a number of direct or indirect reuse opportunities. Community reference FOMs [such as the DIS-based Real-time Platform Reference FOM (RPR FOM)] also provide a source of reuse. Merging federate SOMs or employing reusable FOM components [e.g., *base object models* [BOMs)] also represent efficient means of developing HLA FOMs compared to "clean sheet" approaches. Modern OMDTs generally produce a required initialization file for HLA RTIs (e.g., FED, FDD, or XML files). This is an activity output that is not explicitly identified in the DSEEP.

A.4.4.2 Activity 4.2—Establish federation agreements

There are many different types of agreements that are necessary for any distributed simulation application to operate properly. The general categories of these agreements are articulated in the DSEEP and include such items as the use of common algorithms, common databases, and common initialization procedures. While these general classes of agreements apply across most simulation architectures and methodologies, the content of these agreements may be unique to the HLA. For instance, agreements on synchronization points, save/restore procedures, and ownership transfer are all very dependent on the HLA Interface Specification service selection, which the associated *federation agreements* must reflect. Agreements on roles and responsibilities with respect to publishing/subscribing FOM data should also be part of the federation agreements.

A.4.4.3 Activity 4.3—Implement federate designs

Other than the need to check that the federate HLA interface is sufficiently robust for the application at hand, and the possible need to certify HLA compliance for the federate, this activity is identical to that of the DSEEP.

A.4.4.4 Activity 4.4—Implement federation infrastructure

The implementation of supporting infrastructure (routers, networks, bridges, etc.) is largely independent of the supporting architecture/methodology. Thus, this activity is basically identical to that of the DSEEP. However, possible modifications of the RTI initialization data (RID) file to improve federation performance is unique to HLA applications.

A.4.5 Step 5: Plan, integrate, and test simulation environment

A.4.5.1 Activity 5.1—Plan execution

The planning of the execution and documentation of the execution environment are activities that are required of all architectures and methodologies. Thus, this activity is identical to that of the DSEEP.

A.4.5.2 Activity 5.2—Integrate federation

Integration activities are a necessary part of all simulation environment developments. However, the integration must obviously be tied to the selected methodology and associated protocols. For the HLA, this basically means that all federates are properly installed and interconnected via the RTI in a configuration that can satisfy all FOM data interchange requirements and federation agreements. To accomplish this, some HLA-specific integration tasks will be required, such as installation of the RTI and other HLA-support tools for federation monitoring and data logging.

A.4.5.3 Activity 5.3—Test federation

Several aspects of simulation environment testing are unique to HLA applications. First, the interfaces of each federate must be tested to confirm that it can publish and subscribe to FOM data to the extent necessary for that federate to meet its assigned responsibilities. Such testing also includes the verification that all appropriate federation agreements are being adhered to. Once this level of testing has been accomplished, the federates are then interconnected via the RTI to verify basic connectivity and to demonstrate the ability to exchange information at runtime as defined by the FOM. Finally, the complete federation is examined for semantic consistency and generally to verify that the representational requirements (as defined by the *federation requirements* and *conceptual model*) are being met at the specified level of fidelity. Although each of these three levels of testing is also necessary for non-HLA applications, the HLA paradigm drives the nature of testing tasks at all levels.

A.4.6 Step 6: Execute federation and prepare outputs

A.4.6.1 Activity 6.1—Execute federation

The sequence of actions required to execute an HLA federation is similar to that of other architectures and methodologies. However, depending on the types of federates involved in the federation, there may be certain actions that are unique to the HLA. For instance, the existence of certain federation management tools and data loggers may necessitate a specific join order. Also, the use of synchronization points and the need to periodically save the state of the federation may have implications in the way the execution is conducted. It is also generally beneficial to have HLA-experienced personnel on hand during the execution in case RTI problems or issues with other supporting HLA tools occur.

A.4.6.2 Activity 6.2—Prepare federation outputs

Other than the possible use of specialized HLA tools to assist with the reduction and reformatting of data from HLA data loggers, this activity is identical to that of the DSEEP.

A.4.7 Step 7: Analyze data and evaluate results

A.4.7.1 Activity 7.1—Analyze data

This activity is identical to that described in the DSEEP.

A.4.7.2 Activity 7.2—Evaluate and feedback results

The process of analyzing the data from the federation execution and producing the final results is identical to that of the DSEEP. With respect to archiving reusable products, there are several HLA-specific products that are potentially reusable, such as the FOM and the SOMs of the participating federates. Other potentially reusable products are not necessarily unique to the HLA (e.g., same products are also produced by users of other architectures and methodologies), but the content of these products is HLA-unique. Examples include the federation objectives statement, the federation agreements, and the federation test data.

Annex B

(normative)

Distributed Interactive Simulation process overlay to the DSEEP

B.1 Introduction

Distributed Interactive Simulation (DIS) is a government/industry partnership to define an architecture and supporting infrastructure for linking simulations of various types at multiple locations to create realistic, complex, virtual "worlds" for the simulation of highly interactive activities. This infrastructure brings together systems built for separate purposes, technologies from different eras, products from various vendors, and platforms from various services and permits them to interoperate. DIS exercises are intended to support a mixture of virtual entities with computer-controlled behavior (computer-generated forces), virtual entities with live operators (human in-the-loop simulators), live entities (operational platforms and test and evaluation systems), and constructive entities (war games and other automated simulations).

The basic tenets that underlie the DIS architecture include the following:

- No central computer controls the entire simulation exercise.
- Autonomous simulation applications are responsible for maintaining the state of one or more simulation entities.
- A standard protocol is used for communicating ground truth data.
- Changes in the state of an entity are communicated by simulation applications.
- Perception of events or other entities is determined by the receiving application.
- Dead reckoning algorithms are used to reduce communications processing.

In general, DIS is best suited for real-time simulation environments at the platform-level of resolution. In addition, the application-specific data exchange requirements must be achievable via the DIS *protocol data units* (PDUs) as defined in the standard. For applications that can be characterized in this way, DIS provides an effective and (relatively) easy way to implement an interoperable distributed simulation environment.

The IEEE maintains the current set of DIS specifications: IEEE Std 1278.1™ [B11], IEEE Std 1278.2™ [B12], and IEEE Std 1278.3™ [B13]. These specifications are presently available to users as are many commercial and government tools/utilities to facilitate the development of DIS applications.

B.2 Terminology mappings and definitions (specific architecture/methodology to DSEEP)

B.2.1 Terminology mappings

Simulation environment:	Simulation exercise
Member application:	Simulation application[5]
Simulation data exchange model	See footnote 6.

B.2.2 Definitions

DIS compliant: A simulation that can send or receive PDUs in accordance with IEEE Std 1278.1-1995 [B11] and IEEE Std 1278.2-1995 [B12]. A specific statement must be made regarding the qualifications of each PDU.

entity: Any component in a system that requires explicit representation in a model. Entities possess attributes denoting specific properties. *See also:* **simulation entity.**

exercise: (A) One or more sessions with a common objective and accreditation. **(B)** The total process of designing, assembling, testing, conducting, evaluating, and reporting on an activity. *See also:* **simulation exercise.**

measure of effectiveness (MOE): Measure of how the system/individual performs its functions in a given environment. Used to evaluate whether alternative approaches meet functional objectives and mission needs. Examples of such measures include loss exchange results, face effectiveness contributions, and tons delivered per day.

measure of performance (MOP): Measure of how the system/individual performs its functions in a given environment (e.g., number of targets detected, reaction time, number of targets nominated, susceptibility of deception, task completion time). It is closely related to inherent parameters (physical and structural), but measures attributes of system/individual behavior.

session: A portion of an exercise that is contiguous in wall clock time and is initialized by a session database.

session database: A database that includes network, entity, and environment initialization and control data. It contains the data necessary to start a session.

simulation application: (A) The executing software on a host computer that models all or part of the representation of one or more simulation entities. The simulation application represents or "simulates" real-world phenomena for the purpose of training or experimentation. Examples include manned vehicle (virtual) simulators, computer generated forces (constructive), environment simulators, and computer interfaces between a DIS network and real (live) equipment. The simulation application receives and processes information concerning entities created by peer simulation applications through the exchange of DIS PDUs. More than one simulation application may simultaneously execute on a host computer. **(B)** The

[5] The term *member application* is actually somewhat broader than the term *simulation application.* Other applications such as data loggers or a control station are also types of Member Applications.
[6] DIS does not have a corresponding term for *simulation data exchange model.* However, the set of PDUs that are being used to support a particular DIS simulation exercise are collectively equivalent to the SDEM term as it is used in the DSEEP.

application layer protocol entity that implements standard DIS protocol. The term simulation may also be used in place of simulation application.

simulation entity: An element of the synthetic environment that is created and controlled by a simulation application and is affected by the exchange of DIS PDUs. Examples of types of simulated entities include tank, submarine, carrier, fighter aircraft, missiles, bridges, or other elements of the synthetic environment. It is possible that a simulation application may be controlling more than one simulation entity. *See also:* **entity.**

simulation exercise: An exercise that consists of one or more interacting simulation applications. Simulations participating in the same simulation exercise share a common identifying number called the *exercise identifier*. These simulations also utilize correlated representations of the synthetic environment in which they operate. *See also:* **exercise.**

simulation fidelity: (A) The similarity, both physical and functional, between the simulation and that which it simulates. **(B)** A measure of the realism of a simulation. **(C)** The degree to which the representation within a simulation is similar to a real-world object, feature, or condition in a measurable or perceivable manner.

subject matter expert (SME): Individual knowledgeable in the subject area being trained or tested.

B.3 Global mapping (specific architecture/methodology to DSEEP)

The five-step DIS exercise development and construction process[7] (see IEEE Std 1278.3 [B13]) was designed to guide users through the various activities necessary to build and execute a DIS distributed simulation application. The mapping between this process and the DSEEP is illustrated in Figure B.1. This mapping is also summarized in Table B.1. Note that there is a direct correspondence between each step in the DIS Process and at least one step in the DSEEP. Also note that the fundamental activities required to build and execute the simulation environment are essentially the same between the two processes (although they are packaged somewhat differently), and that the basic sequence of these activities is also the same. Thus, at this global level perspective, there are no differences that preclude a successful implementation of the DSEEP in a DIS context. However, there are obvious differences in terminology, and at the next level of decomposition, other differences begin to emerge with respect to how the generic activities and lower-level tasks described in the DSEEP can be implemented in a DIS context. The following subclauses identify the major steps and activities required to build and execute a DIS simulation exercise, including a summary of the differences with the more generic DSEEP activity descriptions.

[7] Formally known as the *DIS exercise management and feedback process.*

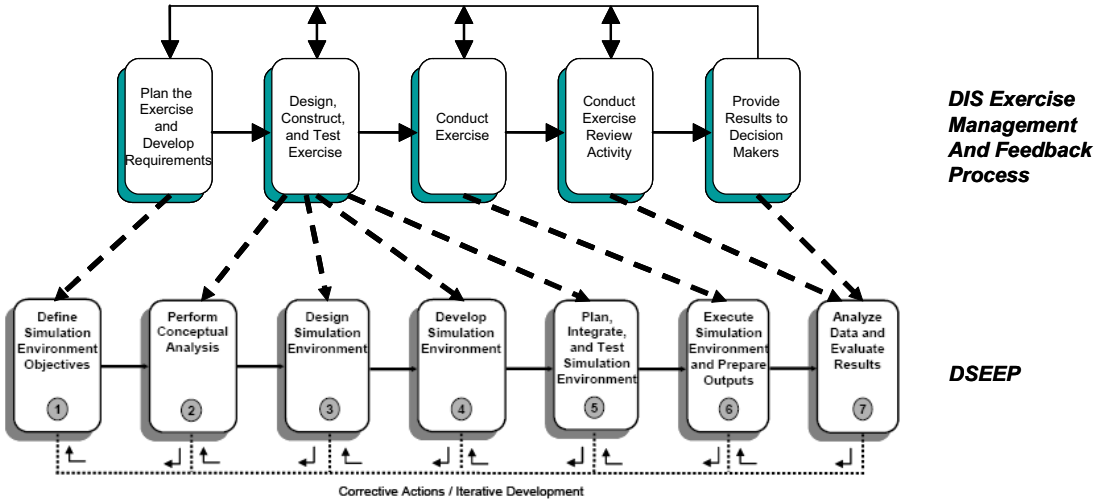

Figure B.1—DIS Process to DSEEP mapping

Table B.1—Top-level mapping of DIS Process to DSEEP

DSEEP / DIS	Define simulation environment objectives	Perform conceptual analysis	Design simulation environment	Develop simulation environment	Integrate and test simulation environment	Execute simulation	Analyze data and evaluate results
Plan the exercise and develop requirements	X						
Design, construct, and test exercise		X	X	X	X		
Conduct exercise						X	
Conduct exercise review activity							X
Provide results to decision maker							X

The DIS Process also describes a set of roles for the personnel, agencies, organizations, or systems involved in the management of any DIS exercise. Since these roles are referenced in the Clause 3 discussions, they are provided here for reference:

User/Sponsor: The DIS User or Sponsor is the person, agency, or organization that determines the need for and scope of a DIS exercise and/or establishes the funding and other resources for the exercise. The

User/Sponsor also determines the exercise participants, objectives, requirements, and specifications. The User/Sponsor appoints the Exercise Manager and VV&A Agent.

Exercise Manager: The Exercise Manager is responsible for creating the exercise, executing the exercise, and conducting the post-exercise activities. The Exercise Manager coordinates with the Verification, Validation and Accreditation (VV&A) Agent during these tasks and then reports the results of the exercise to the User/Sponsor.

Exercise Architect: The Exercise Architect designs, integrates, and tests the exercise as directed by the Exercise Manager.

Mode/Tool Providers: The Model/Tool Providers develop, stock, store, maintain, and issue simulation assets. They maintain historical records of utilization and VV&A.

Site Managers: The Site Managers maintain and operate the physical simulation assets located at their geographic locations. They coordinate with the Model/Tool Providers to install and operate specific simulation and interaction capabilities specified by the Exercise Manager. Sites may include operational live equipment. Incorporation of live equipment requires special coordination.

Network Manager: The Network Manager is responsible for maintenance and operation of a network capable of providing the DIS link between two or more sites. For a given exercise, the Exercise Manager selects a Network Manager.

VV&A Agent: The VV&A Agent is the person, agency, or organization appointed by the User/Sponsor to measure, verify, and report on the validity of the exercise, and to provide data allowing the User/Sponsor to accredit the results.

Exercise Analyst: The Exercise Analyst is the person, agency, or organization tasked to reduce the exercise data and provide analytical support.

Exercise Security Officer: The Exercise Security Officer verifies that the exercise planning, conduct, and feedback are compliant with all applicable laws, regulations, and constraints associated with security of the network and the participating systems (e.g., communications, sites, processing).

Logistics Representative: The Logistics Representative participates in all exercise phases, interfaces with other systems on logistics issues, and evaluates logistics realism, regardless of whether logistics is specifically portrayed in the exercise.

B.4 Detailed mappings (specific architecture/methodology to DSEEP)

B.4.1 Plan the exercise and develop requirements

This first step involves performing a set of critical planning tasks for the DIS exercise. The Exercise Manager leads this activity, but should coordinate with the exercise User/Sponsor and obtain technical support while performing these planning activities. All member application representatives of the Exercise Management Team participate in the planning for the DIS exercise. As a minimum, the Exercise Manager should consider the following:

a) Exercise purpose, objectives, exercise completion date, and security requirements

b) Measures of effectiveness (MOEs) and measures of performance (MOPs) applicable to the exercise

c) The required feedback products, audience, and timeliness

d) Specific data collection requirements [e.g., PDU set, field radio frequency (RF) data set, and other mechanisms to support tasks b) and c)]

e) Plans for VV&A, test, configuration management, etc.

f) Exercise schedule

g) Rules of engagement and political environment

h) Location to be simulated

i) Exercise scenario time frame

j) Exercise environment (weather, climate, electromagnetic, oceanographic)

k) Exercise forces—friendly, opposing, and neutral

l) Mix of simulation forces among live, virtual, and constructive categories

m) Simulation resources available

n) Simulation resources to be developed

o) Technical and exercise support personnel required

p) Applicable range safety requirements

q) Initial conditions, planned events, scenarios, and vignettes

r) Functional/performance requirements and interface specifications for models and instrumented ranges

s) Required utilities to translate data from instrumented ranges and sensors to DIS format

t) Battlespace databases

u) Plan for including logistics realism

v) Contingency plan for reallocation of resources to recover from unavailability of planned simulation assets

In the DSEEP, this first step is divided into three separate activities, reflecting the need to (1) define the high-level sponsor needs, (2) develop a translation of those needs into a concrete set of simulation objectives (to guide early conceptual analysis and detailed requirements development), and (3) produce an initial project plan. In the DIS Process, the task list above reflects the content of all of these basic activities within a single step. As part of this step, the User/Sponsor is assumed to work collaboratively with the development/integration team throughout the planning process, communicating not only the high-level exercise objectives, but also any practical constraints (e.g., resources, schedule, security) that may restrict the exercise development.

Table B.2 provides an approximate mapping between the DIS Step 1 tasks and the activities in DSEEP Step 1. Some observations follow.

Table B.2—DIS Step 1 tasks mapped to DSEEP activities

DIS tasks / DSEEP Step 1	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v
Identify User/Sponsor needs	X	X	X										X									
Develop objectives								X	X	X	X						X					
Initiate planning					X	X									X							X
Post Step 1				2			2					3						2		2		

— There are some DIS Step 1 tasks that have no easily identifiable counterpart in the DSEEP. These are tasks p), s), and u). In each of these cases, these are tasks specific to potential DIS user groups and represent a simple tailoring of the generic guidance provided by the DSEEP.

— There are some DIS Step 1 tasks that are identified by the DSEEP as being performed later in the overall process. These are indicated in the last row of the table, where each number identifies the step in the DSEEP for which the task is described. For instance, rules of engagement and the political environment [DIS task g)] is defined as being part of DSEEP conceptual analysis activities in Step 2, while the specification of data collection requirements [DIS Task d)] is defined as part of DSEEP requirements development activities also in Step 2. Since both the DIS Process and the DSEEP allow for considerable flexibility in the timing of required activities and tasks, this is not considered to be a significant problem.

— While the DIS Process only identifies planning activities as part of Step 1, the DSEEP identifies initial planning activities as part of Step 1, and detailed planning as part of Step 3. However, it is expected that updates to plans developed to support a DIS exercise will be modified as needed as the development progresses, and thus there is no real difference here.

— In terms of products, the main product produced by the first step of the DIS Process is the project plan, which reflects the collective outcome of all the Step 1 tasks. However, the DSEEP has products for all of its Step 1 activities, specifically the *needs statement,* the *objectives statement,* and the initial project plan. This does not mean that the DIS Process does not capture the same information; it simply means that DIS users generally choose to merge the contents of these three DSEEP products into a single product. Note that there is nothing in the DIS Process to preclude users and developers from producing additional products if they wish. The single product is just a reasonable tailoring of generic DSEEP guidance, which can be altered based upon the specific needs of the application.

B.4.2 Design, construct, and test exercise

In this step of the DIS Process, the Exercise Architect develops the DIS exercise to meet the requirements specified during the planning phase, making maximum reuse of existing DIS components. The Exercise Manager, along with the Model/Tool Providers, plays a major role in the design and construction of the exercise. The exercise development includes selection or development of components such as simulation applications, databases, architecture, and environment.

This DIS Process step consists of a sequence of five phases: conceptual design, preliminary design, detailed design, construction and assembly, and integration and testing. The following describes each of these phases, along with its mapping to DSEEP steps or activities.

B.4.2.1 Conceptual design

In this phase, the Exercise Architect develops the conceptual model and top-level architecture for the exercise that shows the participating components, their interfaces, behavior, and control structure. This maps quite closely to Step 2 (perform conceptual analysis) of the DSEEP. However, DSEEP Step 2 focuses mainly on the specification of the domain in software-independent terms, while the corresponding DIS phase begins to extend this into the software domain (to some degree) to more effectively bridge into preliminary design. DSEEP Step 2 also includes additional activities related to detailed requirements and scenario development, which the DIS Process identifies as a Step 1 activity.

B.4.2.2 Preliminary design

In this phase, the Exercise Architect translates the requirements developed during the planning phase into a preliminary DIS exercise. This includes development of scenario(s), mission plans for various participants, database and map development and distribution, communications network design and tests, and planning for training and rehearsals.

This DIS Process phase maps most closely to DSEEP Step 3 (design simulation environment). However, the DSEEP defines only a single simulation environment design effort instead of separate preliminary and detailed design phases. For lower-level differences, see B.4.2.3.

B.4.2.3 Detailed design

In this phase, the Exercise Architect works with the Exercise Manager to elaborate on the design model and architecture generated in the previous step to the extent necessary to support and complete the definition of all required functions, data flow, and behavior. The Exercise Architect and Exercise Manager specifically include communication data rate requirements and data latency limitation requirements.

The preliminary and detailed design phases together map to DSEEP Step 3. However, there are some differences. In the DSEEP, Step 3 is when the exercise participants are selected based on their ability to meet the exercise requirements specified in Step 2. In the DIS Process, selection of participating simulations occurs earlier during project planning, although supplementary (or alternative) exercise participants can always be added as requirements change. DSEEP Step 3 also includes a specific activity to translate the initial project plan into a more detailed plan for development and execution. Although this is not identified as a separate activity in the DIS Process, it is understood that the project plan will continue to be refined as necessary throughout the development process. Finally, this phase of the DIS Process allows for some degree of database development, while this type of development activity is normally considered part of Step 4 in the DSEEP.

B.4.2.4 Construction and assembly

In this phase, the Exercise Manager, with the assistance of the Model/Tool Providers and the Exercise Security Officer, assembles existing DIS components and develops new components that meet all security requirements. This phase maps to DSEEP Step 4. However, DSEEP Step 4 provides a more detailed breakdown of the activities required to construct and assemble an exercise, such as establishing agreements among exercise participants on supporting resources (e.g., data sources, common algorithms), implementing required modifications to individual exercise participants, and establishing the underlying simulation infrastructure. In the DIS Process, such activities are simply assumed to occur as required during construction/assembly.

In Step 4 of the DSEEP, there is also an activity dedicated to the development of the SDEM. Since DIS does not have the concept of an SDEM, there is no mention of such an activity in the DIS Process. However, it is necessary for all participants in a DIS exercise to agree on precisely which PDUs are to be used for a given exercise, whether default or exercise specific values are used, and which enumerations are used. Collectively, this selected set is roughly comparable to the notion of an SDEM as it is used in the DSEEP.

B.4.2.5 Integration and testing

In this phase, the Exercise Manager and Exercise Architect work this activity as an incremental process, starting with a minimum number of components and connectivity, then adding and building until they reach operational status. They then test to determine whether or not requirements and performance criteria are met. The exercise support personnel are also trained and rehearsed (dry run).

This phase maps very closely to DSEEP Step 5. Like the preceding phase, the DSEEP describes this activity in more depth, placing a special emphasis on the planning aspects of integration and test. However, the basic series of tasks is essentially the same between the two processes. In fact, the incremental nature of the DIS testing process as previously described maps very well with the three-layered testing approach (member application testing, integration testing, interoperability testing) described in the DSEEP.

B.4.3 Conduct exercise

In this step of the DIS Process, the Exercise Manager conducts the DIS exercise using the resources developed during the design, construct, and test phase. The goal of this third phase is to satisfy the established objectives. The Exercise Manager is responsible for all necessary management functions.

This step of the DIS Process maps to DSEEP Step 6. The DSEEP is more explicit about the tasks required to confirm proper execution management and runtime data collection, but it is understood that these same tasks occur in DIS exercises. The DSEEP also identifies a specific activity for post-processing raw exercise outputs. The purpose of this activity is primarily data reduction, although translation into desired formats may also be necessary in preparation for data analysis activities in the next step.

B.4.4 Conduct exercise review activity

In this step of the DIS Process, a review of the exercise is conducted. The exercise review activity may initially center on rapidly providing after-action review material. The Exercise Manager can recall events or situations marked with observed event markers in the data log and incorporate the data into presentation material. Topics about which the Exercise Manager may wish to draw material for analysis and feedback include interactions, communications, rules of engagement, logistics, and command decisions. From this material, analysts and exercise participants can attempt to develop an understanding of the exercise and of the decision-making processes based on what each participant knew and perceived at a given time. The

Exercise Analysts may want to obtain tools to select, integrate, and display exercise data for these purposes. These data can also be supplemented by information provided by exercise observers and analysts located with the exercise participants during the conduct of the exercise. The Exercise Manager must work with the Exercise Architect who will task the Model/Tool Providers to develop tools to select, integrate, and display exercise data. Additionally, the Exercise Manager verifies that exercise data is archived for future cross-exercise analysis.

The exercise feedback system should include various data presentation methods. The exercise feedback system should also provide a selection of analytical functions for preliminary analysis and after-action review of each exercise, as well as support the bulk of post-exercise data consolidation and analysis tasks.

This step of the DIS Process maps to the first activity (analyze data) within DSEEP Step 7. The DSEEP provides less detail than the DIS Process for this activity, as the rather concise, generic guidance provided by the DSEEP has been translated into more detailed instructions for DIS users. However, at the task level, the processes are entirely consistent.

B.4.5 Provide results to decision-makers

In this step of the DIS Process, exercise results are reported to designated levels of User/Sponsor and other audiences according to the reporting requirements of the exercise. These results may include the following:

- Exercise credibility
- Cause and effect relationships
- Detail and aggregation
- Analysis
- Exercise improvement

This step of the DIS Process maps to the second activity (evaluate and feedback Results) within DSEEP Step 7. While the DIS Process is more specific about the classes of results that can be provided to the User/Sponsor, the DSEEP includes an additional task for archiving all reusable products produced throughout the development/execution process. In general, the DSEEP is more explicit about identifying such products, and is the reason why the reusability of these products is emphasized more strongly in the DSEEP.

Annex C

(normative)

Test and Training Enabling Architecture process overlay to the DSEEP

C.1 Introduction

Test and Training Enabling Architecture (TENA) is designed to enhance software interoperability and reuse throughout the range community. The architecture gives guidance on how to design range software applications such that they can easily interact with other TENA applications to support a range event. The architecture also specifies a common TENA object model, akin to a common set of interface definitions, which enables this interoperability through a community-wide understanding of range-related information. The primary benefit of the TENA architecture is the ability it gives exercise planners to rapidly compose TENA-compliant applications into a geographically distributed "logical range" for a range event. A logical range integrates testing, training, simulation, and high-performance computing technologies (distributed across many facilities) and ties them together with a common architecture. TENA is sponsored by the Office of the Secretary of Defense (OSD) Central Test and Evaluation Investment Program (CTEIP).

The TENA architecture includes a technical *concept of operations* (ConOps) for planning, creating, testing, and using a logical range (see U.S. DoD [B15]). This ConOps represents a broad overview of range operations, with explicit descriptions of where TENA brings capabilities or requirements. The content of this process is based on extensive feedback from range engineers as obtained through Common Test and Training Range Architecture (CTTRA) workshops and other such interactions.

The TENA Logical Range ConOps is described in terms of three sequential phases, that of *pre-event, event,* and *post-event* (see Figure C.1). Across these three phases, there are five major activities, as follows:

— *Requirements definition*—The purpose and objectives of the range event are defined.

— *Event planning*—All of the planning is completed for the event including the definition of the scenario.

— *Event construction, setup, and rehearsal*—All of the physical preparation for the event is done, including software development, LROM creation, hardware and communication system setup, and testing.

— *Event execution*—The scenario is played out and data is collected.

— *Analysis and reporting*—Collected data is analyzed with respect to the event objectives and reports are generated.

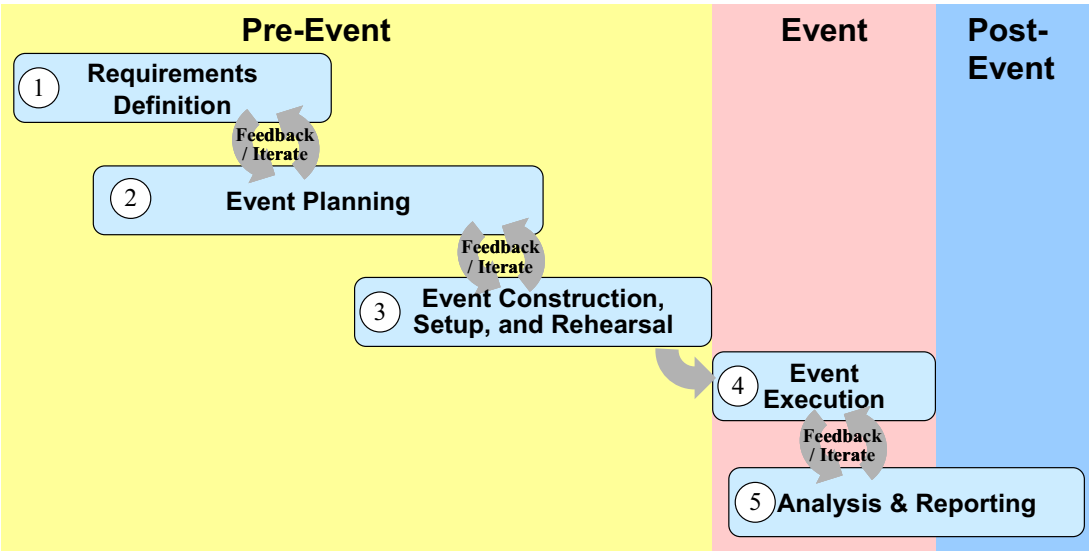

Figure C.1—TENA Logical Range ConOps

Each activity in this figure has a purpose, a list of information inputs, a set of subactivity, and a list of products. Like the DSEEP, the activities (along with the subactivity within each activity) can be performed iteratively and/or concurrently. The TENA Logical Range ConOps also defines explicit roles that can be assumed by logical range participants, and how those roles relate to the various activities. Collectively, the ConOps provides a broad and tailorable view of the development and execution activities used by range engineers to support a TENA-based range event.

The U.S. DoD maintains the current set of TENA specifications and are presently available to users (see U.S. DoD [B15]), as are many commercial and government tools/utilities to facilitate the development of TENA applications.

C.2 Terminology mappings, acronyms, and definitions (specific architecture/ methodology to DSEEP)

C.2.1 Terminology mappings

Simulation environment:	Logical range
Member application:	Range resource application
Simulation data exchange model:	Logical range object model (LROM)

C.2.2 Acronyms and abbreviations

AAR	after-action review
COI	critical operational issue
ConOps	concept of operations
CTEIP	Central Test and Evaluation Investment Program

DSEEP	Distributed Simulation Engineering and Execution Process
KPP	key performance parameter
LROM	logical range object model
LVC	live, virtual, constructive
MOE	measure of effectiveness
MOP	measure of performance
SUT	system under test
T&E	test and evaluation
TENA	Test and Training Enabling Architecture
V&V	verification and validation

C.2.3 Definitions

The following identifies a number of key definitions for terms used in this annex.

logical range: A suite of TENA resources, sharing a common object model, that work together for a given range event.

logical range object model (LROM): Those object definitions, derived from whatever source, that are used in a Logical Range execution to meet the immediate needs and requirements of a specific user for a specific range event.

measures of effectiveness (MOE): Measures designed to correspond to accomplishment of mission objectives and achievement of desired effects.

measures of performance (MOP): A criterion used to assess friendly actions that are tied to measuring task accomplishment.

Test and Training Enabling Architecture (TENA): A software architecture designed to enhance and enable interoperability, reuse, and composability of software resources among the test and training, range, and simulation communities. TENA software includes the TENA Middleware and a set of tools and utilities to enable users to create and maintain live, virtual, constructive (LVC) environments known as *logical ranges.*

C.3 Global mapping (specific architecture/methodology to DSEEP)

As discussed above, the TENA Logical Range ConOps consists of five major activities spread among three sequential phases. The mapping between the ConOps and the top-level view of the DSEEP is illustrated in Figure C.2. This mapping is summarized in Table C.1.

Note that there is a direct correspondence between each activity in the ConOps and at least one step in the DSEEP. Also note that the fundamental activities required to build and execute a distributed M&S environment are essentially the same between the two processes (although they are structured somewhat differently), and with only a few minor exceptions, the basic sequence of these activities is also the same. Thus, at this global level perspective, there are no major differences that preclude a successful implementation of a logical range using the DSEEP. However, there are obvious differences in terminology, and at the activity/subactivity level, other differences begin to emerge with respect to how the generic activities and lower-level tasks described in the DSEEP can be implemented for TENA applications. The following subclauses identify the major steps and activities required to construct and

execute a TENA-based range event, including a summary of the differences with the more generic DSEEP activity descriptions.

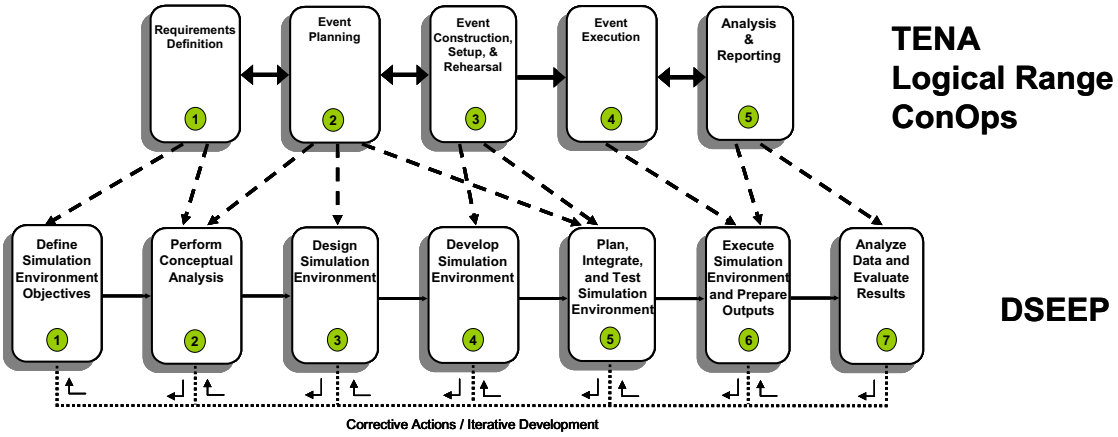

Figure C.2—TENA Logical Range ConOps to DSEEP mapping

Table C.1—Top-level mapping of TENA Logical Range ConOps to DSEEP

DSEEP / ConOps	Define simulation environment objectives	Perform conceptual analysis	Design simulation environment	Develop simulation environment	Integrate and test simulation environment	Execute simulation	Analyze data and evaluate results
Requirements definition	X	X					
Event planning		X	X		X		
Event construction, setup, and rehearsal				X	X		
Event execution						X	
Analysis and reporting						X	X

C.4 Detailed mappings (specific architecture/methodology to DSEEP)

The purpose of this subclause is to define the mapping between the subactivities in the TENA Logical Range ConOps and corresponding activities in the DSEEP. This expands upon the previous top-level mapping by presenting one additional level of decomposition. A summary of the textual description of this mapping (as provided throughout this subclause) is given in Table C.2.

C.4.1 Requirements definition

The first activity in the TENA Logical Range ConOps focuses on elaborating customer-defined requirements, defining the overall purpose/objective of the range event, and iterating specific requirements and goals based on feasibility considerations. This activity includes an assessment as to what mission capability is required, and based on that assessment, identification of the high-level scenario (including the necessary tactical systems and operational threads). The requirements and scenario drive a logical range conceptual model that maps and decomposes the scenario operational context into segments (physical entities, environment entities, organization entities, hierarchy, etc.). A high-level analysis plan is also produced that defines how conclusions will be drawn from the event, what data is required to support obtaining those conclusions, and how the data obtained will be analyzed.

The first subactivity in ConOps Activity 1 is to *define mission and mission support requirements*. The focus is to identify programmatic and operational constraints such as schedule drivers, security issues, equipment availability, and cost. Critical operational issues (COIs), measures of effectiveness (MOEs), measures of performance (MOPs), and key performance parameters (KPPs) are also identified. This subactivity is completely analogous to the DSEEP identify user/sponsor needs activity in Step 1.

The second subactivity in ConOps Activity 1 is to *establish event objectives*. The purpose of this subactivity is to formally document the objectives of the event. In the DSEEP, the objectives statement is an output of the develop objectives activity, also in Step 1.

The third subactivity in ConOps Activity 1 is an analysis process entitled *develop high-level scenario*. In this subactivity, the necessary forces, tactical systems, equipment, operational threads, and operational environment are all identified. All entities and their essential behaviors are described, including the relationships between the entities that allow the scenario to be played out. This subactivity develops a fairly coarse specification of the scenario representation that aligns with aspects of both the DSEEP Step 1 develop objectives activity and the early stages of the DSEEP Step 2 develop scenario activity.

The fourth subactivity in ConOps Activity 1 is to *perform logical range conceptual analysis*. The objective of this activity is to develop an implementation-independent representation of the event. The event representation is linked to the event objectives so that traceability can be maintained between the customer's needs and the high-level scenario. In the DSEEP, this subactivity corresponds to the develop conceptual model activity in Step 2. Also, in terms of products, the output of this subactivity (logical range conceptual model) is simply a test community-specific instantiation of the generic conceptual model produced as a result of this DSEEP activity.

The fifth subactivity in ConOps Activity 1 is to *define event requirements*. The objective is to accumulate and document the results of the previous steps into a formal requirements document, including the mission requirements (COIs, MOEs, MOPs, KPPs), event objectives, scenario, conceptual model, and all of the derived requirements that have been identified. In the DSEEP, the development of formal requirements is performed in the develop simulation environment requirements activity as a key component of Step 2.

Table C.2—Activity-level mapping of TENA Logical Range ConOps to DSEEP

TENA ConOps \ DSEEP	Identify User/Sponsor Needs	Develop Objectives	Conduct Initiate Planning	Develop Scenario	Develop Conceptual Model	Develop Simulation Environment Requirements	Select Member Applications	Design Simulation Environment	Prepare Detailed Plan	Develop Simulation Data Exchange Model	Establish Simulation Environment Agreements	Implement Member Application Designs	Implement Simulation Environment Infrastructure	Plan Execution	Integrate Simulation Environment	Test Simulation Environment	Execute Simulation Environment	Prepare Simulation Environment Outputs	Analyze Data	Evaluate and Feedback Results
	Step 1			Step 2			Step 3			Step 4				Step 5			Step 6		Step 7	
Activity 1																				
Define Mission and Mission Support Requirements	X																			
Establish Event Objectives		X																		
Develop High-Level Scenario		X		X																
Perform Logical Range Conceptual Analysis					X															
Define Event Requirements						X														
Create Analysis Plan, Cost and Schedule Estimates			X																	
Activity 2																				
Determine Resources Needed							X	X												
Research Prior Event Information							X	X												
Develop Detailed Event Schedule									X					X						
Develop Detailed Scenario				X																
Allocate Functionality								X												
Analyze Logical Range Concept and Define the Detailed Description of the Logical Range								X												
Establish Detailed Event Procedures and Plans									X					X						
Activity 3																				
Define Logical Range Object Model										X										
Implement Upgrades												X								
Create Initialization Data											X									
Set Up and Test the Logical Range													X		X	X				
Address Contingencies																X				
Rehearse Event																X				
Activity 4																				
Initialize Event																	X			
Control and Monitor Range Resources																	X			
Execute Scenario																	X			
Capture and Archive Data																	X			
Manage and Monitor Logical Range																	X			
Assess Ongoing Event																	X			
Activity 5																				
Generate Quick-Look Reports/Hot-Washes																	X			
Consolidate Collected Data																		X		
Post-Process and Reduce Data																			X	
Replay the Test Mission/Debrief the Training Exercise																				X
Deposit New TENA Resources in the TENA Repository																				X
Generate Event Report/Execute After Action Review and Create Take-Home Package																				X
Document, Distribute, and Archive "Lessons Learned" in TENA Repository																				X

The sixth subactivity in ConOps Activity 1 is to *create analysis plan, cost and schedule estimates*. This task finishes the documentation for the requirements definition activity by creating the Analysis Plan, a blueprint for how to achieve the objectives through the analysis process, as well as an estimate of the event cost and schedule. This subactivity maps to the conduct initial planning activity in DSEEP Step 1, although the Analysis Plan will include many more T&E-specific tasks than are articulated in the generic planning documents defined in the DSEEP document.

C.4.2 Event planning

The focus of the second activity defined in the TENA Logical Range ConOps is to create detailed plans for: the event execution; scenario entities, events, and time-lines; supporting range resources; analysis operations; and data collection. All plans necessary for creating the event are finalized in this activity, including detailed plans for the scenario and the event cost and schedule.

The first subactivity in ConOps Activity 2 is *determine resources needed*. In this subactivity, the list of resources necessary for the completion of the range event is formalized. Resources include human participants, computers, software, networks, range resource applications, systems-under-test, and/or the training audience. This subactivity must take into consideration resource constraints, including availability, cost, security restrictions, realism and fidelity, quality, and risk management. The determination of required resources is performed in two different DSEEP Step 3 activities, that of select member applications and design simulation environment. More specifically, the select member applications activity is concerned with identifying LVC simulation assets that can meet functional requirements, while the design simulation environment activity is where supporting hardware/software resources (such as computers, networks, etc.) are identified. Together, these two DSEEP activities produce what is identified as the output of this single ConOps subactivity.

The second subactivity in ConOps Activity 2 is to *research prior event information*. Each previous range event may have information (such as scenario information, collected data, or lessons learned) that may be useful in creating the new range event. Leveraging this information should make the planning process for new logical range events more efficient. Like in the previous subactivity, this maps most closely to the select member applications and design simulation environment activities of DSEEP Step 3. This mapping is more implicit than explicit, in that both of these DSEEP activities emphasize the reuse of existing resources, and existing resources can only be reused if information on prior applications can be obtained via mining appropriate data repositories and other available information stores.

The third subactivity in ConOps Activity 2 is *develop detailed event schedule*. The purpose here is to develop the detailed schedule for the range event. If the event is taking place only at a single range, the scheduling process will go forward much as it does today. If the event is a multi-range event, the event must be cooperatively scheduled among all range resource owners. Mutually acceptable time periods must be coordinated, including time for check-out and verification of resources, network and component integration and testing, event rehearsal, as well as time for the actual event execution. In the DSEEP, detailed event planning is split between two separate activities. First, the develop detailed plan activity in Step 3 produces a coordinated plan for the development, test, and execution of the simulation environment, including the specific tasks and milestones for each member application. Then, closer to the actual event, the plan is refined and augmented in the plan execution activity in Step 5 with the final details of the execution environment and additional operational planning considerations (e.g., required personnel, rehearsal, and execution schedule). Although this is shown as a single subactivity in the ConOps, it is understood that plan refinements can occur at any stage of ConOps Activity 2 and 3 implementation.

The fourth subactivity in ConOps Activity 2 is *develop detailed scenario*. This subactivity produces a more detailed refinement of the original high-level scenario, including the military forces and operational concepts, the time-lines of events that make up the larger test and/or training event, and the roles of any range resources being used. This subactivity maps to the develop scenario activity in DSEEP Step 2, which produces the same basic product.

The fifth subactivity in ConOps Activity 2 is *allocate functionality*. This subactivity determines which of the selected resources produces data and/or collect data throughout the event. In the DSEEP, the allocation of required functionality to the assets in the simulation environment is performed as an integral part of the design simulation environment activity in DSEEP Step 3.

The sixth subactivity in ConOps Activity 2 is *analyze logical range concept and define the detailed description of the logical range*. Here, the logical range developers do a complete analysis of their logical range concept. The centerpiece of this analysis is the development and simulation of the logical range's "information architecture"—a detailed description of the range resource applications in their roles as information providers and consumers, as well as the physical networks associated with a given logical range configuration. This analysis determines that the desired configuration is technically viable under the assumptions and requirements dictated by the scenario. In the DSEEP, this same information is produced in the design simulation environment activity of Step 3. More specifically, the element of the simulation environment design identified as "simulation environment architecture (including supporting infrastructure design and standards to be supported)" in the DSEEP is equivalent to the product produced as a result of this ConOps activity.

The seventh subactivity in ConOps Activity 2 is to *establish detailed event procedures and plans*. This involves finalizing all of the detailed plans related to an event, including the security procedures (both physical and network), communications agreements, interface control documents, range safety plans, operating procedures, detailed test procedures or training orders of battle, resource memoranda of agreements, configuration management plans, environmental impact analyses, and the final event staffing plan. Like the previous develop detailed event schedule subactivity, this corresponds to two different DSEEP activities; that of the develop detailed plan activity in Step 3 and the plan execution activity in Step 5. As previously explained, the DSEEP captures this planning information in a sequence of plan refinements rather than in a single location within the broader process. These types of plan refinements occur in all simulation developments; however, it is explicit in the DSEEP while being implicit in the ConOps.

C.4.3 Event construction, setup, and rehearsal

The focus of the third activity defined in the TENA Logical Range ConOps is to create the physical pre-conditions for the event. Unlike the event planning activity that produces paper products, the products of this activity are software applications, databases, and configurations of range resources. As part of this activity, the LROM is defined, any range resource applications are upgraded to support this LROM, and the logical range configuration is integrated, tested, rehearsed, and made ready for event execution.

The first subactivity in ConOps Activity 3 is *define logical range object model*. This step defines the LROM that will be used in this event. This LROM may be entirely reused from previous events, may have slight changes from ones used in previous events, or may be entirely new based on new requirements. The set of existing object definitions for the chosen range resource applications is generally the starting point for the development of the LROM. Any conflicts between mutually incompatible object definitions are then resolved and a unified set of classes for the LROM defined. This subactivity corresponds to the develop simulation data exchange model activity in Step 4 of the DSEEP. Note that while the DSEEP emphasizes reuse of existing modeling resources, it does not assume that object definitions (as the basis of simulation data exchange) are provided as part of the selected simulation architecture. TENA does provide a set of object definitions in the specifications, although this set can be extended at any time based on user requirements.

The second subactivity in ConOps Activity 3 is to implement upgrades. These upgrades to the range resource applications could be for the purpose of adding new functionality, updating algorithms, or integrating with new range hardware. The upgrades could also be for integrating any changes necessary because of new object definitions described in the LROM. In the DSEEP, this maps very closely to the implement member application designs activity in Step 4. Such changes are generally due to member

application upgrades necessary to conform to the end-product of the establish simulation environment agreements activity.

The third subactivity in ConOps Activity 3 is to *create initialization data*. Each TENA range resource application requires some information to start successfully, and this subactivity is to build the required initialization databases. Examples of this information include scenario information, synthetic environment information, and operating parameters. In the DSEEP, the development of initialization data is performed in the establish simulation environment agreements activity in Step 4. One of the key tasks identified as part of this DSEEP activity is to "build all required databases," which includes the required initialization data.

The fourth subactivity in ConOps Activity 3 is to *set up and test the logical range*. In this subactivity, the hardware, software, databases, and networks that comprise the logical range are assembled into a system and tested to make sure they can communicate and operate as intended. Generally, portions of the logical range are set up and tested before the entire logical range is assembled to give confidence that the portions are working before a full system test. This includes the calibration of any instrumentation. This subactivity maps to three different activities in the DSEEP. First, the required hardware and software infrastructure elements (e.g., computers, routers, bridges) that enable communication among distributed member applications is installed and tested as part of the implement simulation environment infrastructure activity in Step 4. Then, the member applications are interconnected into a unified operating environment in the integrate simulation environment activity in Step 5. Finally, the simulation environment undergoes testing at several levels to verify proper operation prior to execution. This final step corresponds to the test simulation environment activity in DSEEP Step 5.

The fifth subactivity in ConOps Activity 3 is *address contingencies*. This subactivity is intended to encompass all problem-solving activities that occur before an event that are not explicitly planned for, but are necessary to get the logical range's hardware, software, instrumentation, and networks to function properly. In the DSEEP, addressing such contingencies is assumed to be part of the testing process, and thus is addressed in the test simulation environment activity in DSEEP Step 5.

The sixth subactivity in ConOps Activity 3 is *rehearse event*. In test events, rehearsals of varying degrees are performed in which some aspects (such as the system-under-test) are replaced with simulations. Training events too may have rehearsals of varying fidelity and scope. The rehearsal step is very similar to, and indeed could entirely overlap with, the logical range tests described in the set up and test the logical range subactivity previously discussed. The DSEEP also recognizes this overlap, and thus includes all pre-execution testing events and resulting corrective actions as part of the test simulation environment activity in Step 5.

C.4.4 Event execution

The focus of the fourth activity defined in the TENA Logical Range ConOps is to execute the event based on the plans created in the event planning activity and with the range resources, databases, and networks created and integrated in the event construction, setup, and rehearsal activity. The necessary data is collected and some real-time/quick-look analysis is performed.

The following provides a short description of the six subactivity performed in Activity 4:

a) *Initialize event*—All logical range resources read in their initialization information and are brought into their fully operational state.

b) *Control and monitor range resources*—Each range resource application is controlled and monitored during the event to detect and resolve any application failures.

c) *Execute scenario*—The scenario is executed according to plan. Each military unit or system, whether as part of a test or training exercise, acts according to its plans and any feedback from the exercise conductors.

d) *Capture and archive data*—All relevant data is captured, based on the data collection plan developed in the earlier create analysis plan, cost, and schedule estimates subactivity.

e) *Manage and monitor logical range*—The logical range as a whole is managed, monitored, and adjusted to make sure it is meeting the customer's objectives.

f) *Assess ongoing event*—An ongoing assessment of the event is performed to confirm it is progressing according to plan and will meet the customer's objectives.

All six of these subactivity map to the single execute simulation activity in DSEEP Step 6. The subactivity breakout described in the ConOps provides a more detailed view into how this one DSEEP activity is implemented in a testing and/or training context.

C.4.5 Analysis and reporting

The focus of the fifth activity defined in the TENA Logical Range ConOps is to provide detailed review and analysis of the event execution and the data collected during the execution. For a test event, the analysis provides answers to the fundamental questions that the test was designed to address. For a training event, a substantial amount of training value is received during the event execution activity itself. However, the analysis activity produces important feedback for both the training audience and the other event participants, and thus enhances the training value of the event.

The first subactivity in ConOps Activity 5 is *generate quick-look reports/hot-washes*. This subactivity is performed while the event is still executing. The basis for this analysis may be data that is collected by specialized real-time data analysis applications or by real-time queries. Like all of the various subactivity described in ConOps Activity 4, this subactivity maps to the execute simulation activity in DSEEP Step 6. That is, the quick-look reports and hot-washes produced during execution are all considered part of execution monitoring, which the DSEEP identifies as a key aspect of the execution process.

The second subactivity in ConOps Activity 5 is to *consolidate collected data*. In many cases, the event data will be collected at multiple geographically separated points throughout the logical range. Efficient analysis is only possible when this data is consolidated at a central location. This consolidation process, which may take some time, must be performed before extensive detailed analysis may be done. In the DSEEP, all preprocessing of raw simulation data is performed in the prepare simulation environment outputs activity in Step 6. In particular, this corresponds to the "merge data from multiple sources" task within this activity.

The third subactivity in ConOps Activity 5 is to *post-process and reduce data*. In this subactivity, data is analyzed to determine how the system-under-test performed or how the training audience acted, and to identify what significant or important trends occurred during the event. Tools that provide sophisticated data mining, pattern recognition, visualization, and statistical analysis techniques are used to assist in processing the data and drawing the appropriate conclusions. In the DSEEP, this maps to the analyze data activity in Step 7. This activity includes a task to "apply analysis methods and tools to data," which is expanded upon in the document text to discuss the same types of data reduction and post-processing actions as this ConOps activity identifies.

The final four subactivity in ConOps Activity 5 are as follows:

a) *Replay the test mission/debrief the training exercise*—Replay/summarize significant portions of the event for the purpose of getting a better understanding of what took place.

b) *Deposit new TENA resources in the TENA repository*—Any new reusable software (including range resource applications, gateways, and tools) is submitted to the TENA repository for possible reuse in future logical ranges.

c) *Generate event report/execute after action review and create take-home package*—For test events, the final event report, containing a summary and analysis of all of the information relevant to the purpose of the event, is written and distributed to the customer. For a training exercise, an *after-action review* (AAR) is held for the training audience for the purposes of reinforcing the training they received during the event and generating lessons learned. Also for a training exercise, a take-home package, analogous to the final report of a test event, is generated.

d) *Document, distribute, and archive "lessons learned" in TENA repository*—Archive lessons-learned in a user-friendly format in the TENA repository so that future logical range designers can benefit from the problems that have been studied and solved in previous logical ranges.

These four subactivity collectively map to the Evaluate and Feedback Results activity in DSEEP Step 7. In particular, the "feedback results" part of this DSEEP activity is mainly concerned with facilitating proper feedback to all stakeholders, which would include the same debriefs, AARs, and archiving of reusable products described in these ConOps subactivity.

Annex D

(informative)

Bibliography

[B1] Scrudder R., et al., "Graphical Presentation of the Federation Development and Execution Process," Simulation Interoperability Workshop, Fall 1998.

[B2] SISO-STD-001.1, Realtime Platform Reference (RPR) FOM, August 1999.

[B3] SISO-STD-003, Base Object Model (BOM) Template Specification, May 2006.

[B4] Tucker, W., et al., "A Glossary of Modeling and Simulation Terms for Distributed Interactive Simulation (DIS)," August 1995.

HLA IEEE 1516

[B5] IEEE Std 1516™, IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA)—Framework and Rules.[8, 9]

[B6] IEEE Std 1516.1™, IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA)—Federate Interface Specification.

[B7] IEEE Std 1516.2™, IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA)—Object Model Template (OMT) Specification.

HLA Version 1.3

[B8] High Level Architecture (HLA) Interface Specification, Version 1.3, U. S. Department of Defense, April 1998.

[B9] High-Level Architecture (HLA) Object Model Template (OMT) Specification, Version 1.3, U. S. Department of Defense, April 1998.

[B10] High Level Architecture (HLA) Rules, Version 1.3, U. S. Department of Defense, April 1998.

DIS

[B11] IEEE Std 1278.1™-1995, IEEE Standard for Distributed Interaction Simulation—Application Protocols.

[B12] IEEE Std 1278.2™-1995, IEEE Standard for Distributed Interaction Simulation—Communication Services and Profiles.

[B13] IEEE Std 1278.3™, IEEE Recommended Practice for Distributed Interaction Simulation—Exercise Management and Feedback.

[B14] SISO-REF-010, Enumeration and Bit Encoded Values for use with Protocols for Distributed Interactive Simulation Applications, May 2006.

[8] IEEE publications are available from the Institute of Electrical and Electronics Engineers, 445 Hoes Lane, Piscataway, NJ 08854, USA (http://standards.ieee.org/).
[9] The IEEE standards or products referred to in Annex D are trademarks owned by the Institute of Electrical and Electronics Engineers, Incorporated.

TENA

[B15] U.S. Department of Defense, “The Test and Training Enabling Architecture (TENA) Architecture Reference Document,” Version 2002, Nov. 2002.