NASA CONSTELLATION DISTRIBUTED SIMULATION MIDDLEWARE TRADE STUDY

David Hasan
L-3 Communications
Houston, TX
(512) 892-7099
david.hasan@l-3com.com

James D. (Dan) Bowman
Nancy Fisher
Teledyne Brown Engineering
300 Sparkman drive
Huntsville, AL 35806
256-726-1000
dan.bowman@tbe.com
nancy.fisher@tbe.com

Dannie Cutts
AEgis Technologies Group
631 Discovery Drive
Huntsville, AL 35806
256-922-0802
dcutts@aegistg.com

Edwin Z. (Zack) Crues, Ph.D.
2101 NASA Parkway, ER7
NASA Johnson Space Center
Houston, Texas 77058, USA
(281) 483.2902
edwin.z.crues@nasa.gov

Keywords:
NASA, Simulation, Middleware, HLA, TENA, DIS, XML, XMPP, Trade Study.

ABSTRACT: *This paper presents the results of a trade study designed to assess three distributed simulation middleware technologies for support of the NASA Constellation Distributed Space Exploration Simulation (DSES) project and Test and Verification Distributed System Integration Laboratory (DSIL). The technologies are the High Level Architecture (HLA), the Test and Training Enabling Architecture (TENA), and an XML-based variant of Distributed Interactive Simulation (DIS-XML) coupled with the Extensible Messaging and Presence Protocol (XMPP). According to the criteria and weights determined in this study, HLA scores better than the other two for DSES as well as the DSIL.*

1. Introduction

This report presents the results of a study of three middleware technology alternatives for two distributed simulation user communities in the NASA Constellation Program.

The simulation communities considered are the *Distributed Space Exploration Simulation* (DSES) project the *Distributed System Integration Laboratories* (DSIL) and are summarized in the appendix in Section 5.1. The candidate technologies considered are the *High Level Architecture* (HLA), the *Test and Training Enabling Architecture* (TENA), and a combination of an XML-based version of the *Distributed Interactive Simulation standard* and the *Extensible Messaging and Presence Protocol* (DIS-XML/XMPP). These candidates are summarized in the appendix in Section 5.3.

The specific question addressed by this report is "Which of these three middleware technologies is best suited use for use in DSES and DSIL?"

Notice that the scope of the study is on the evaluation of these three technologies against each other. In particular, it does not consider the question of whether some other approach is preferable (e.g., custom development of the distributed simulation middleware) and it does not address questions concerning DSES and DSIL simulation architecture. For example, concerns about how to accommodate the communications latencies due to geographical separation of time-sensitive components in the DSES or DSIL are not in the scope of this study. Our focus was exclusively on the relative merits of HLA, TENA and DIS-XML/XMPP.

The following sections of this report address the study's *method* (the process by which the middleware candidates were assessed), the study *results* (the criteria, raw scores and weighted grades assigned to each middleware candidate as applied to DSES and DSIL) and the study's *conclusions* (which technology is best suited for DSES and which for DSIL).

2. Methodology

The method employed by this study to evaluate the middleware candidates is based on the *Analytic Hierarchy Process* (AHP). [1,2] The process is an approach to selecting between different alternatives using qualitative as well as quantitative criteria. It is a structured way of assigning scores to the various criteria and using numerical weights to assess the application of the various candidates to specific contexts, each of which has its own set of weights. The process involves the following steps.

- Specify the evaluation criteria.
- Determine relative criterion weights for each application context.
- Assign raw scores to the criteria for each candidate.
- Use the scores and weights to grade the candidates in each application context.

2.1 Specifying the Criteria

AHP calls for the decomposition of the problem into a hierarchical set of categories against which to score the candidates. At the "bottom" of this hierarchical decomposition are the specific criteria against which the candidates are scored. For example, one possible category could be *overall performance*, which might be decomposed into various subcategories including *network performance*, which in turn might be decomposed into two criteria: *latency minimization* and *throughput maximization*.

In this study, the criteria were decomposed into three general categories: *user operations*, *implementation performance* and *programmatic considerations*. Examples of user operation criteria include checkpointing, synchronization points and global event ordering. Examples of implementation performance include network latency and throughput. Examples of programmatic considerations include training costs and whether or not the middleware is based on an open and/or international standard. The appendix in Section 5.3 presents the criterion hierarchy, although a detailed description of each of the categories is beyond the space available for this paper.

2.2 Determining the Relative Weights

For each application context (e.g., DSES and DSIL), AHP calls for the determination of a set of numerical weights expressed as percentages that specify the relative significance of the various criteria as applied to specific application contexts. These weights are determined by balancing the importance of the criteria in a particular category against each other. For example, for a *network performance* category consisting of the criteria *latency minimization* and *throughput maximization*, AHP would force the analyst to decide whether latency is more significant than throughput in each context.

In this study, two sets of weights were generated: one for DSES and one for DSIL. The three top-level categories, user *operations*, *implementation performance* and *programmatics*, were assigned relative weights of (53%, 31%, 16%), respectively for both contexts. However, at deeper levels of the hierarchy, the DSES and DSIL weights differed. For example, the performance category was decomposed further into four subcategories, *responsiveness*, *efficiency*, *robustness* and *scalability*, which were assigned relative weights (51%, 11%, 27%, 12%) and (63%, 12%, 20%, 5%) for the DSES and DSIL contexts, respectively. A detailed discussion of the AHP mechanism by which these were determined is beyond the scope of this paper, but all the weights used in this study are presented in Appendix 4.

2.3 Assigning Raw Scores

In AHP, each candidate (e.g., HLA, TENA, and DIS/XML) is scored with respect to each criterion. This assessment consists of assigning numerical scores to each of the criteria for that candidate. These are not absolute numbers but rather relative scores that quantify how the candidates perform *relative to each other* with regard to each criterion. In other words, the candidates are considered one pair at a time, and raw scores are selected that reflect how the first candidate compares to the second with respect to the relevant criterion. For three candidates, that means three raw scores for each criterion.

The numerical values of these raw scores are based on the following standard AHP values. (In the event that the second candidate scores better than the first, the reciprocal of these values is used.)

Score	***Meaning***
1	Both candidates are equivalent.
2	Between 1 and 3.
3	1st candidate is slightly better than the 2nd.
4	Between 3 and 5.
5	1st candidate is strongly better than the 2nd.
6	Between 5 and 7.
7	1st candidate is very strongly better than the 2nd.
8	Between 7 and 9.
9	1st candidate is overwhelmingly better than the 2nd.

Table 2.3.1 *Score Meanings*

These raw scores are then normalized so that the pairwise scores for a particular criterion add to 100. The normalized scores are used in the subsequent grading process.
Since there are three candidates (HLA, TENA and DIS/XML) in this study, three pairwise scores were determined for each criterion, one for each of the following pairs:

- HLA compared to TENA,
- HLA compared to DIS/XML and
- TENA compared to DIS/XML.

For example, for the availability of online help criterion, the normalized scores were:

Raw	*Norm.*	*comment*
1/2	32	HLA has the DoD Modeling and Simulation Information Analysis Center (MSIAC).
3	63	TENA has a very powerful online and phone help desk available through the TENA web site.
5	6	No known support available for DIS or DIS/XML, although some online support is available for open source XMPP servers.

Table 2.3.2 *Example Scores*

The scores for all the criteria are summarized in Section 5.4.

2.4 Grading the Candidates

AHP provides a structured method of summing the normalized criterion scores for all the criteria in order to derive a grade for each application context. Context-specific grades are obtained by multiplying the scores by context-specific weights and summing these products to derive an overall context-specific grade. The details are slightly more involved than this, since the hierarchical nature of the criterion decomposition involves several levels of weights, some applied to categories and sub-categories and others applied to the criteria themselves. See the references for more information on this. Introductory AHP information is also readily available online.

In this study, since there were three middleware candidates (HLA, TENA and DIS/XML) and two application contexts (DSES and DSIL), there are six overall grades: one for each of the candidates in each of the contexts. These grades are shown below.

Context	*Grades*		
	HLA	*TENA*	*DIS/XML*
DSES	45.0	32.3	22.7
DSIL	44.0	32.4	23.6

Table 2.4.1 *Overall Grades*

The highest score in each row indicates the best middleware candidate for that application context.

3. Conclusions

As can be seen in the table above, the overall grade for HLA is the highest of the three candidates in both application contexts. Indeed, after finding these results, an additional sensitivity analysis was conducted to investigate to what extent this result is sensitive to changes in the raw scores or changes in the DSES and DSIL weights. The results of this analysis suggest that modest changes to the scores and weights do not lead to different results.

To the extent that the criteria, weights and scores used in this study are reasonable, the conclusion of this study is that HLA is the best middleware candidate of the three considered.

4. Acknowledgements

The authors would like to thank the following people for their assistance. Danny Thomas and Bobby Hartway, both of the AEgis Technologies Group, helped kick off this study and conduct the AHP analysis. The management and staff at the TENA Software Development Activity (SDA) organization provided invaluable help in building TENA benchmark applications. Jim Gibson of SAIC ported HLA v1.3 Defense Modeling and Simulation Office (DMSO) benchmarks to HLA/IEEE-1516. Finally, the NASA Constellation Program provided the funding for this work.

5. Appendices

5.1 Simulation Communities

This appendix describes the two application contexts considered by this study: the Distributed Space Exploration Simulation (DSES) and the Distributed System Integration Laboratory (DSIL).

DSES. The Distributed Space Exploration Simulation (DSES) project is sponsored by the Constellation System Engineering and Integration (SE&I) Modeling and Simulation Data Architecture (MSDA) Office. It focuses on technologies and processes related to high fidelity, collaborative, interoperable (and optionally distributed) simulation of the Constellation system of systems architecture (e.g., Crew Launch Vehicle (CLV), Crew Exploration Vehicle (CEV), etc.). The project uses the development of Constellation-related simulations to begin developing an understanding of the infrastructure and technologies necessary to pursue this vision.

Current DSES simulation federates interact with each other as a High Level Architecture (HLA) federation. HLA is an IEEE standard that provides a general framework within which simulation developers can structure and describe their simulation applications. HLA addresses two key issues: promoting interoperability between simulations and aiding the reuse of models in different contexts. The DSES simulation has used HLA to demonstrate simulations built and run from geographically separated locations; however, the real benefit of the DSES infrastructure is not so much this ability to deploy simulations in a distributed fashion but rather the interoperability that comes from designing them as if they were.

The DSES project has used distributed simulations to drive several technology areas: development of a software infrastructure to promote distributed and interoperable simulations, initial development of a distributed simulation network, and demonstration of Constellation capabilities through the rapid integration of domain experts at various NASA centers.

DSIL. The Distributed System Integration Laboratory (DSIL) will consist of multiple Constellation System Integration Labs (SILs), interacting with each other over a broadband network to provide the capability to test (a subset of) Level 2 requirements (including interfaces among Constellation systems, and possibly integrated Constellation performance, etc.). Additionally, since some Constellation system-system interactions cannot realistically be tested in all cases in a geographically distributed fashion (primarily due to the latencies of communication in relation to the time constant of closed loop interactions inherent in Constellation system design), additional system HWIL representation may be physically co-located at one or more other SILs (e.g., CLV flight processor HWIL configuration physically located at the CEV SIL for example) to provide a configuration to be able to test these tightly coupled interactions for these additional cases.

Currently the DSES and DSIL projects are collaborating in a build approach for DSIL that maximally leverages the experience, expertise, and capabilities developed to date as part of the DSES project to the development of the DSIL capability to accomplish the following objectives:

- avoid duplication (and therefore reduce Constellation costs)
- help increase fidelity of distributed simulation for Constellation
- develop an architecture in which SILs will be interchangeable with simulations at the Constellation level; and avionics-based components within a given systems SIL will be interchangeable with a software model of that component

5.2 Middleware Candidates

The following sections describe the three candidate technologies considered in this study: HLA, TENA and DIS-XML/XMPP. The three candidates are

- HLA/IEEE-1516, an IEEE standard version of the High Level architecture
- the DoD-based Test and Training Enabling Architecture (TENA), and
- DIS-XML, a combination of Distributed Interactive Simulation (DIS), the Extensible Markup Language (XML), and the Extensible Messaging and Presence Protocol (XMPP).

Other distributed computing technologies exist (e.g., CORBA and Jini), but this study has focused on these three candidates because of their direct relevance to the distributed simulation context relevant to the Constellation DSES and DSIL projects.

HLA. HLA [3,4] originated in the United States Department of Defense (DoD) as a standard set of services for linking distributed simulations and training applications. It was eventually standardized as IEEE-1516. HLA does not specify on-the-wire data representations. It does specify a set of rules that distributed simulations ("federates") must obey in order to form a legal HLA "federation" and a set of services (with C++ and Java mappings) through which the simulations interact with each other and the HLA runtime infrastructure. There are several commercial HLA implementations.

TENA. TENA [5,6] also originated in the United States Department of Defense (DoD). It was designed to support interoperability and reuse among DoD test and training ranges. It provides an object-oriented approach for real-time exchange of data and invocation of remotely located objects. The DoD Central Test and Evaluation Investment Program (CTEIP) sponsors TENA middleware development and distributes the only implementation.

DIS/XML. DIS [7] is an on-the-wire protocol defined by IEEE standard 1278. It was developed based on experience with the Simulation Networking (SIMNET) Advanced Research Projects Agency (ARPA) program. DIS is intended to provide an interoperability infrastructure for joining distributed simulations of various types. Much of DIS interoperability comes from the best practices and lessons learned in years of distributed DIS training activities. DIS-XML [8] utilizes the Extensible Markup Language (XML) to encode DIS data on the wire. The advantage of the use of XML is the wide availability of XML-processing tools (in particular, in the Java community). This is particularly relevant to data architects who have an interest in ensuring that all data (perhaps even intermediate results) can be archived in a format that may be meaningfully analyzed later. Although not explicitly intended for distributed simulations, the eXtensible Messaging and Presence Protocol (XMPP) [9,10] chat room concept can be effectively used as a communications mechanism for distributed simulations. XMPP emerged from the open source Jabber instant messaging community.

5.3 Criteria

Table 5.3.1 describes the criteria hierarchy used in this study. Rows in the table that correspond to hierarchical *categories* are shaded. Unshaded rows correspond to *criteria* against which the middleware candidates were measured.

Categories / Criteria	***Meaning***
1. User Operations	Category for criteria that capture tools, capabilities and support.
1.1 Capabilities	Category for criteria that capture the capabilities of the middleware.
1.1.1 Pre-execution	Category for capabilities that apply before the simulation executes.
1.1.1.1 Software engineering process	Category for capabilities related to software engineering processes.
1.1.1.1.1 Defined process	Does the middleware have a built-in set of support tools that aid or enforce the systems engineering documentation process?
1.1.1.1.2 Configuration management	Does the middleware have a built-in set of support tools that aid or enforce the configuration management process?
1.1.1.1.3 Versioning	Does the architecture have a built-in set of support tools that aid or enforce the versioning control process?
1.1.1.2 Type checking	Can the middleware perform data type checking at compile time rather than during integration?
1.1.2 Execution	Category for criteria related to simulation execution.
1.1.2.1 Execution management	Category for criteria related to coordinating a running simulation.
1.1.2.1.1 Checkpointing	Does the middleware support creation of and resuming from checkpoints?
1.1.2.1.2 Synchronization points	Does the middleware support globally coordinated synchronization points?
1.1.2.2 Publication/Subscription	Does the middleware support publish/subscribe?
1.1.2.3 Object ownership	Does the middleware support object ownership?
1.1.2.4 Repeatability	Does the middleware support repeatable simulations?
1.1.2.5 Data filtering	Does the middleware support dynamic, class-based data filtering?
1.1.2.6 Data Transmission	Category for criteria related to data transmission.
1.1.2.6.1 Data streaming	Does the middleware support continuous data streams (e.g., video)?
1.1.2.6.2 Best effort data delivery	Does the middleware support best effort data delivery?
1.1.2.6.3 Reliable data delivery	Does the middleware support guaranteed data delivery?
1.1.2.7 Distribution transparency	Does the middleware support the data transmission without requiring producers and consumers being aware of each other?
1.1.2.8 Object orientation	Does the middleware support "true" object-oriented modeling such as the ability to invoke methods on objects?
1.1.2.9 Global event ordering	Does the middleware support consistent event ordering for all simulation participants?
1.1.3 Post-execution	Category for criteria related to post-simulation activities.
1.1.3.1 Data archiving	Does the middleware support saving and archiving data generated during simulation?
1.1.3.2 Data analysis	Does the middleware support analysis of archived data (e.g., data mining and troubleshooting)?
1.2 Tools	Category for criteria related to middleware-specific tools.
1.2.1 Execution planning & setup	Category for criteria related to pre-execution tools.
1.2.1.1 Object modeling tools	Are tools available to support building, modifying and maintaining object models?
1.2.1.2 Simulation development tools	Are tools available to support the development and maintenance of simulations (e.g., IDEs)?
1.2.2 Execution monitoring and control	Category for criteria related to tools for use during the simulation execution.
1.2.2.1 Data visualization tools	Are tools available to view data during execution?
1.2.2.2 Data recording tools	Are tools available to record runtime data for logging or troubleshooting purposes?
1.2.2.3 Simulation monitoring & control	Are tools available to support runtime monitoring and control of the simulation?
1.2.3 Post-execution tools	Category for criteria related to post-execution tools.
1.2.3.1 Data analysis tools	Does the middleware have data analysis tools?
1.2.3.2 Data archiving tools	Does the middleware have data archiving tools?

1.3 Support	Category for criteria related to middleware support.
1.3.1 Issue reporting	Is an issue reporting process available for the middleware?
1.3.2 Online help / help desk	Is online and/or help desk support available?
1.3.3 Training	Is training available?
2. Implementation Performance	Category for criteria that capture middleware execution.
2.1 Responsiveness	Category for criteria related to middleware responsiveness.
2.1.1 Latency	How much network latency does the middleware create?
2.1.2 Throughput	How much data throughput does the middleware enable?
2.1.3 Concurrent executions/federations	Does the middleware support multiple, concurrent simulation executions?
2.2 Efficiency	Category for criteria related to resource usage.
2.2.1 CPU utilization	How does the middleware consume CPU time?
2.2.2 Memory utilization	How does the middleware consume memory?
2.3 Robustness	Category for criteria related to recovery from faults.
2.3.1 Middleware crash recovery	How well does the middleware tolerate middleware infrastructure crashes?
2.3.2 Network fault recovery	How well does the middleware tolerate network faults?
2.3.3 Simulation/federate crash recovery	Does the middleware provide mechanisms to recover from simulation crashes?
2.4 Scalability	How well does the middleware scale?
3. Programmatic Considerations	Category for criteria that capture programmatic realities.
3.1 Standards	Category for criteria related to middleware standards.
3.1.1 Open architecture	Is the middleware based on an open architecture?
3.1.2 International standard	Is the middleware based on an international standard?
3.1.3 Organization for standard evolution	Is there an standards evolution organization with open membership?
3.1.4 Object model support	Is there standard object model support?
3.2 Costs	Category for criteria related to middleware costs.
3.2.1 Implementation costs	Category for criteria related to acquiring, learning and using the middleware.
3.2.1.1 Middleware	Is the middleware inexpensive?
3.2.1.2 Standard	Are the relevant standards inexpensive?
3.2.1.3 Training and maintenance	Is training and maintenance of simulations based on the middleware inexpensive?
3.2.2 Incorporation of other models	Is it relatively easy to integrate external models into a simulation built on the middleware?
3.2.3 Migration to a different architecture	Is it relatively easy to migrate from this middleware to another architecture?
3.3 Maturity	Category for criteria related to how much "shelf life" the middleware has.
3.3.1 Longevity	Has the middleware been around for a while?
3.3.2 Community of practice	Has a community developed around the middleware?

Table 5.3.1 *Description of the Criteria*

5.4 Scores and Weights

The following tables list the approximate normalized scores and context-specific weights associated with the criteria presented above. Note: scores and weights are only relevant to the criteria themselves and not to the hierarchical categories. Accordingly, the cells for category scores and weights are empty.

Categories / Criteria	*Normalized Scores*		
	HLA	***TENA***	***DIS / XML***
1. User Operations			
1.1 Capabilities			
1.1.1 Pre-execution			
1.1.1.1 Software eng'g process			
1.1.1.1.1 Defined process	61	29	10
1.1.1.1.2 Configuration mgmt	33	33	33
1.1.1.1.3 Versioning	40	40	20
1.1.1.2 Type checking	10	61	29
1.1.2 Execution			
1.1.2.1 Execution management			
1.1.2.1.1 Checkpointing	71	14	14
1.1.2.1.2 Synchronization points	54	8	38
1.1.2.2 Publication/Subscription	45	45	9
1.1.2.3 Object ownership	63	10	27
1.1.2.4 Repeatability	63	27	10
1.1.2.5 Data filtering	64	24	12
1.1.2.6 Data Transmission			
1.1.2.6.1 Data streaming	14	71	14
1.1.2.6.2 Best effort data delivery	40	20	40
1.1.2.6.3 Reliable data delivery	45	45	9
1.1.2.7 Distribution transparency	33	33	33
1.1.2.8 Object orientation	14	71	14
1.1.2.9 Global event ordering	71	14	14
1.1.3 Post-execution			
1.1.3.1 Data archiving	33	33	33
1.1.3.2 Data analysis	33	33	33
1.2 Tools			
1.2.1 Execution planning & setup			
1.2.1.1 Object modeling tools	41	50	9
1.2.1.2 Simulation dev't tools	37	49	14
1.2.2 Monitoring and control			
1.2.2.1 Data visualization tools	33	33	33
1.2.2.2 Data recording tools	33	33	33
1.2.2.3 Sim. monitoring & control	61	29	10
1.2.3 Post-execution tools			
1.2.3.1 Data analysis tools	33	33	33
1.2.3.2 Data archiving tools	29	57	14
1.3 Support			
1.3.1 Issue reporting	32	57	11
1.3.2 Online help / help desk	32	57	11
1.3.3 Training	40	40	20
2. Implementation Performance			
2.1 Responsiveness			
2.1.1 Latency	33	33	33
2.1.2 Throughput	57	32	11

2.1.3 Concurrent executions/federations	40	40	20
2.2 Efficiency			
2.2.1 CPU utilization	40	40	20
2.2.2 Memory utilization	33	33	33
2.3 Robustness			
2.3.1 Middleware crash recovery	57	32	11
2.3.2 Network fault recovery	57	32	11
2.3.3 Simulation/federate crash recovery	24	12	64
2.4 Scalability	57	32	11
3. Programmatic Considerations			
3.1 Standards			
3.1.1 Open architecture	43	43	14
3.1.2 International standard	45	9	45
3.1.3 Organization for standard evolution	33	33	33
3.1.4 Object model support	14	43	43
3.2 Costs			
3.2.1 Implementation costs			
3.2.1.1 Middleware	11	37	52
3.2.1.2 Standard	24	64	12
3.2.1.3 Training and maintenance	24	12	64
3.2.2 Incorporation of other models	12	24	64
3.2.3 Migration to a different architecture	33	33	33
3.3 Maturity			
3.3.1 Longevity	63	30	7
3.3.2 Community of practice	64	23	13

Table 5.4.1 *Normalized Scores*

Categories / Criteria	***Weights (%)***	
	DSES	***DSIL***
1. User Operations		
1.1 Capabilities		
1.1.1 Pre-execution		
1.1.1.1 Software engineering process		
1.1.1.1.1 Defined process	3.9	3.9
1.1.1.1.2 Configuration management	1.0	1.0
1.1.1.1.3 Versioning	1.0	1.0
1.1.1.2 Type checking	2.0	2.0
1.1.2 Execution		
1.1.2.1 Execution management		
1.1.2.1.1 Checkpointing	1.5	0.8
1.1.2.1.2 Synchronization points	1.5	2.3
1.1.2.2 Publication/Subscription	2.9	3.0
1.1.2.3 Object ownership	0.6	0.5
1.1.2.4 Repeatability	4.6	3.2
1.1.2.5 Data filtering	1.3	1.7
1.1.2.6 Data Transmission		
1.1.2.6.1 Data streaming	0.5	0.5
1.1.2.6.2 Best effort data delivery	2.4	2.4
1.1.2.6.3 Reliable data delivery	2.4	2.4
1.1.2.7 Distribution transparency	1.3	1.3
1.1.2.8 Object orientation	0.7	0.9
1.1.2.9 Global event ordering	4.7	5.4
1.1.3 Post-execution		
1.1.3.1 Data archiving	1.8	1.8
1.1.3.2 Data analysis	0.9	0.9
1.2 Tools		
1.2.1 Execution planning & setup		
1.2.1.1 Object modeling tools	1.7	1.7
1.2.1.2 Simulation development tools	0.6	0.6
1.2.2 Execution monitoring and control		
1.2.2.1 Data visualization tools	3.3	3.3
1.2.2.2 Data recording tools	1.2	1.2
1.2.2.3 Simulation monitoring & control	4.4	4.4
1.2.3 Post-execution tools		
1.2.3.1 Data analysis tools	0.7	0.7
1.2.3.2 Data archiving tools	1.4	1.4
1.3 Support		
1.3.1 Issue reporting	1.9	1.9
1.3.2 Online help / help desk	1.0	1.0
1.3.3 Training	1.9	1.9
2. Implementation Performance		
2.1 Responsiveness		
2.1.1 Latency	6.3	12.2
2.1.2 Throughput	6.3	5.9
2.1.3 Concurrent executions/federations	3.1	1.2
2.2 Efficiency		
2.2.1 CPU utilization	2.2	2.4
2.2.2 Memory utilization	1.1	1.2
2.3 Robustness		
2.3.1 Middleware crash recovery	1.2	1.0
2.3.2 Network fault recovery	2.3	1.7
2.3.3 Simulation/federate crash recovery	4.7	3.6
2.4 Scalability	3.7	1.6
3. Programmatic Considerations		
3.1 Standards		
3.1.1 Open architecture	1.6	1.6
3.1.2 International standard	1.6	1.6
3.1.3 Organization for standard evolution	0.9	0.9
3.1.4 Object model support	2.4	2.4
3.2 Costs		
3.2.1 Implementation costs		
3.2.1.1 Middleware	0.4	0.4
3.2.1.2 Standard	0.1	0.1
3.2.1.3 Training and maintenance	0.8	0.8
3.2.2 Incorporation of other models	1.4	1.4
3.2.3 Migration to a different architecture	0.5	0.5
3.3 Maturity		
3.3.1 Longevity	3.2	3.2
3.3.2 Community of practice	3.2	3.2

Table 5.4.2 *DSES and DSIL Weights*

5. References

[1] Navneet Bhashan and Kanwal Rai: "Strategic Decision Making: Applying the Analytic Hierarchy Process," Springer, 2004.

[2] Thomas L. Saaty: "Decision Making for Leaders: The Analytic Hierarchy Process for Decisions in a Complex World," Analytic Hierarchy Process Series, Vol. 2, RWS Publications, 1999.

[3] Simulation Interoperability Standards Committee (SISC) of the IEEE Computer Society: "IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) – Federate Interface Specification," Technical Report IEEE-1516.1-2000, The Institute of Electrical and Electronics Engineers, 2000.

[4] Simulation Interoperability Standards Committee (SISC) of the IEEE Computer Society: "IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) – Framework and Rules," Technical Report IEEE-1516-2000, The

Institute of Electrical and Electronics Engineers, 2000.
[5] J. Russell Noseworthy: “The TENA Middleware: Supporting Real-Time Application Development for the DoD Range Community,” http://www.omg.org/news/meetings/workshops/RT_2003_Manual/Presentations/2-2_Noseworthy.pdf, available as of 24 January 2008.
[6] United States Joint Forces Command: “TENA Software Development Activity (SDA) Web Site,” https://www.tena-sda.org/display/intro/Home, available as of 24 January 2008.
[7] Wikipedia: “Distributed Interactive Simulation,” http://en.wikipedia.org/wiki/Distributed_Interactive_Simulation, available as of 24 January 2008.
[8] Web3D Consortium: “X3D DIS-XML Working Group,” http://www.web3d.org/x3d/workgroup/dis/, available as of 24 January 2008.
[9] The Internet Society: “IETF RFC 3920, Extensible Messaging and Presence Protocol (XMPP): Core, http://www.xmpp.org/rfcs/rfc3920.html, available as of 24 January 2008.
[10] The Internet Society: “IETF RFC 3921, Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence,” http://www.xmpp.org/rfcs/rfc3921.html, available as of 24 January 2008.

Author Biographies

DAVID HASAN is an engineer working for L-3 Communications in Houston, TX. He has developed software for the NASA Mission Control Center, GPS-based autonomous navigation systems, and aerospace simulations. He is currently working on distributed simulation projects in the NASA Constellation Program.

JAMES D. (DAN) BOWMAN is a Senior Systems Analyst at Teledyne Brown Engineering. He supports NASA's Modeling and Simulation / Data Architecture Office for the Constellation program in the integrated management of models and simulations. He leads the adaptation of the MAVERIC simulation to serve as the CLV federate in the HLA-based DSES configuration. He has over twenty years of experience in general aerospace modeling and simulation activities, in addition to real-time hardware-in-the-loop testing.

NANCY FISHER has over 20 years of experience in software development, including development of communications, network management, and hardware-in-the-loop test systems. She is a Senior Systems Analyst at Teledyne Brown Engineering and currently supports development of NASA's DSES CLV federate.

DANNIE CUTTS is a Senior Computer Scientist and Certified Modeling and Simulation Professional (CMSP) with The AEgis Technologies Group Inc. supporting the National Aeronautics and Space Administration (NASA) and the U.S. Joint Forces Command (USJFCOM). He has over 20 years experience in M&S for NASA and the DoD and has been involved with the High Level Architecture (HLA) since 1995 serving on the Interface Specification and Time Management Working Groups. He has provided HLA Training, Cadre support for DMSO, and currently serves on the IEEE Drafting Group for the HLA IEEE 1516 standard. Mr. Cutts has supported numerous federation development efforts as well as projects bringing legacy and new simulations to HLA Compliance.

EDWIN Z. (Zack) CRUES, Ph.D. has supported the Automation, Robotics and Simulation Division at NASA Johnson Space Center for the past 15 years. He has been a member of the Simulation and Graphics Branch, since 2004, where he leads the research and development of distributed simulation technologies. In this capacity, he leads the development of NASA's Integrated Mission Simulation (IMSim) which was formerly the Distributed Space Exploration Simulation (DSES). The IMSim work is in support of the Modeling and Simulation Laboratories office for the Constellation program.

2008 Spring Simulation Interoperability Workshop

NASA Constellation Distributed Simulation Middleware Trade Study

April 2008

David Hasan - L-3 Communications
Nancy Fisher & James (Dan) Bowman - Teledyne-Brown
Dannie Cutts & Bobby Hartway - AEgis Technologies
Edwin (Zack) Crues - NASA Johnson Space Center

TELEDYNE
Brown Engineering, Inc.
A Teledyne Technologies Company

outline

- **background**
- **objectives**
- **methodology**
- **results**
- **conclusions**

background

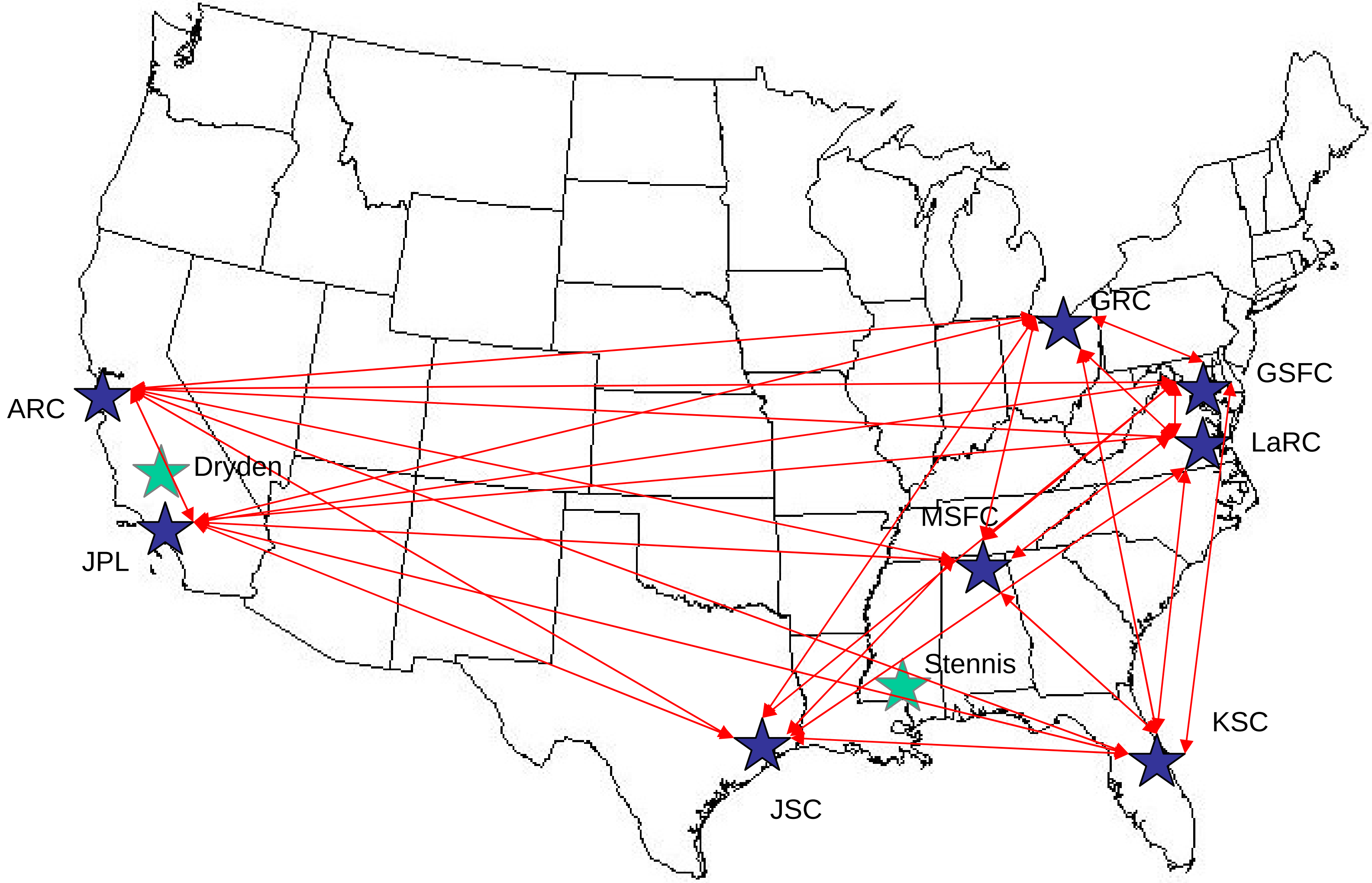
genesis

- **JSC/Trick**
  - master/slave
  - real-time
- **NASA/JAXA ISS-HTV trainer**
  - distributed flight controller trainer
  - Texas-Japan
  - HLA-based
- **simulation-based acquisition**
  - Trick presentation
  - ISS-HTV capabilities
  - token funding
  - JSC, LaRC, ARC
  - infrastructure & proof-of-concept
- **Distributed Space Exploration Simulation (DSES)**
  - JSC: crew vehicle
  - LaRC: launch abort system
  - ARC: crew-triggered abort
  - MSFC: booster

DSES

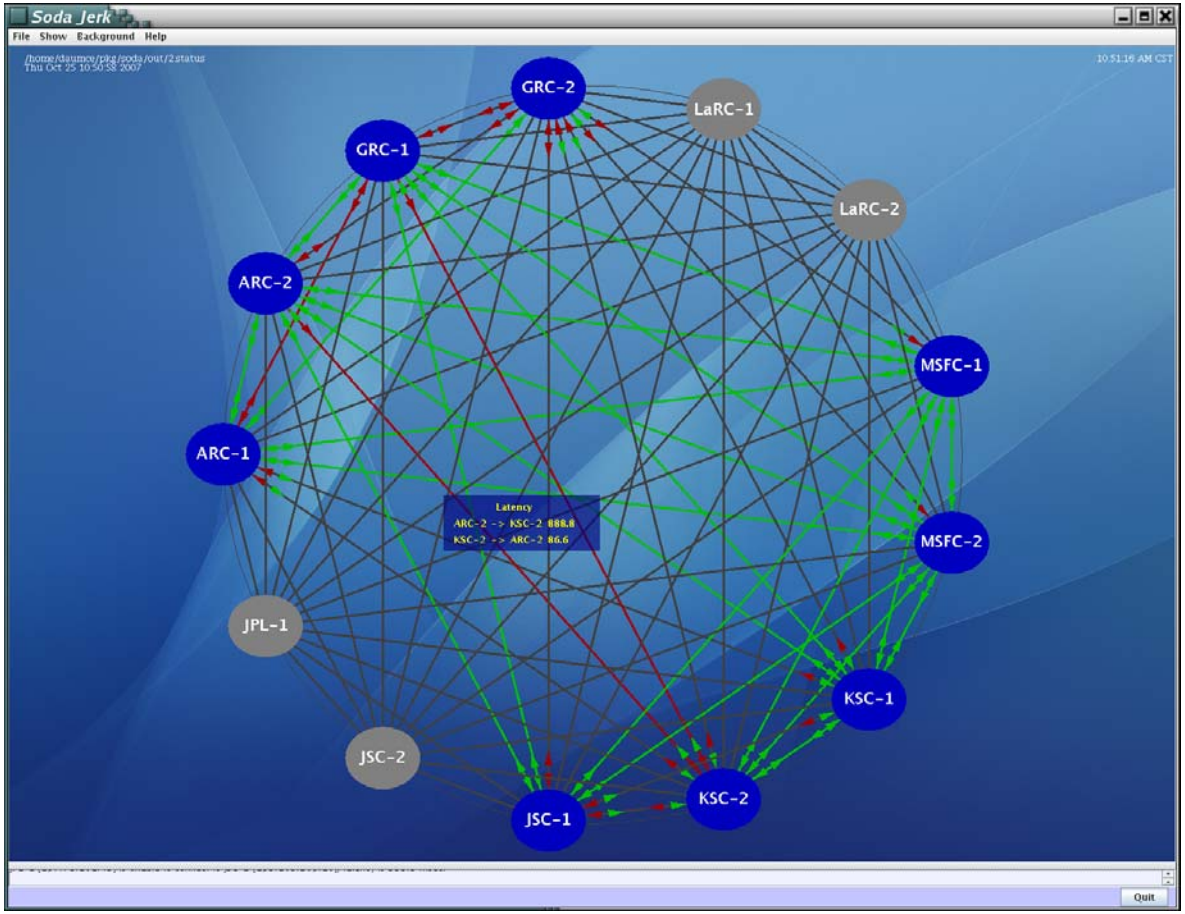
- **focus:**
  - infrastructure
  - expertise
  - products (distributed Orion/Ares-I simulations)
- **infrastructure & expertise:**
  - FOM
  - HLA development and deployment
  - coordinated firewall rules
  - software tools (Trick/HLA interface)
- **simulation capabilities:**
  - pre-launch (mobile launcher at pad)
  - launch & ascent
  - abort (optional)
  - ISS rendezvous & docking
- **DSES is now IMSim**

DSNet



- **background**
- objectives
- methodology
- results
- conclusions

network monitoring



- **background**
- objectives
- methodology
- results
- conclusions

Orion/Ares-I simulation

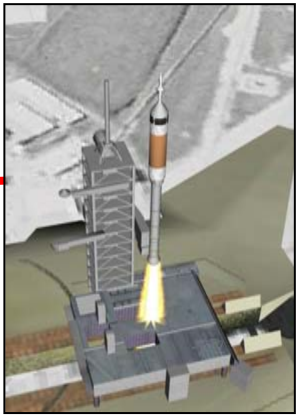
MSFC

JSC

ARC

LaRC

KSC



Integrated Distributed Orion/Ares Simulation

NASA DSNet

- **background**
- objectives
- methodology
- results
- conclusions

Distributed System Integration Laboratory (DSIL)

- **software and avionics test and verification**
- **System Integration Laboratories (SILs)**
- **emulators**
- **distributed testing**

middleware

- **what have we been using?**
  - HLA / IEEE-1516

- **alternatives?**
  - TENA
  - DIS
  - DIS/XML
  - CORBA
  - Jini
  - sockets
  - reflective memory

future

- **DSES (IMSim)**
  - end-to-end flight simulation
  - comm & tracking network simulation
  - mission rehearsal simulation

- **DSIL**
  - demonstration of new capabilities
  - risk reduction
  - distributed test & verification

objectives

middleware candidates

- HLA
- TENA
- DIS/XML
- no others

two contexts

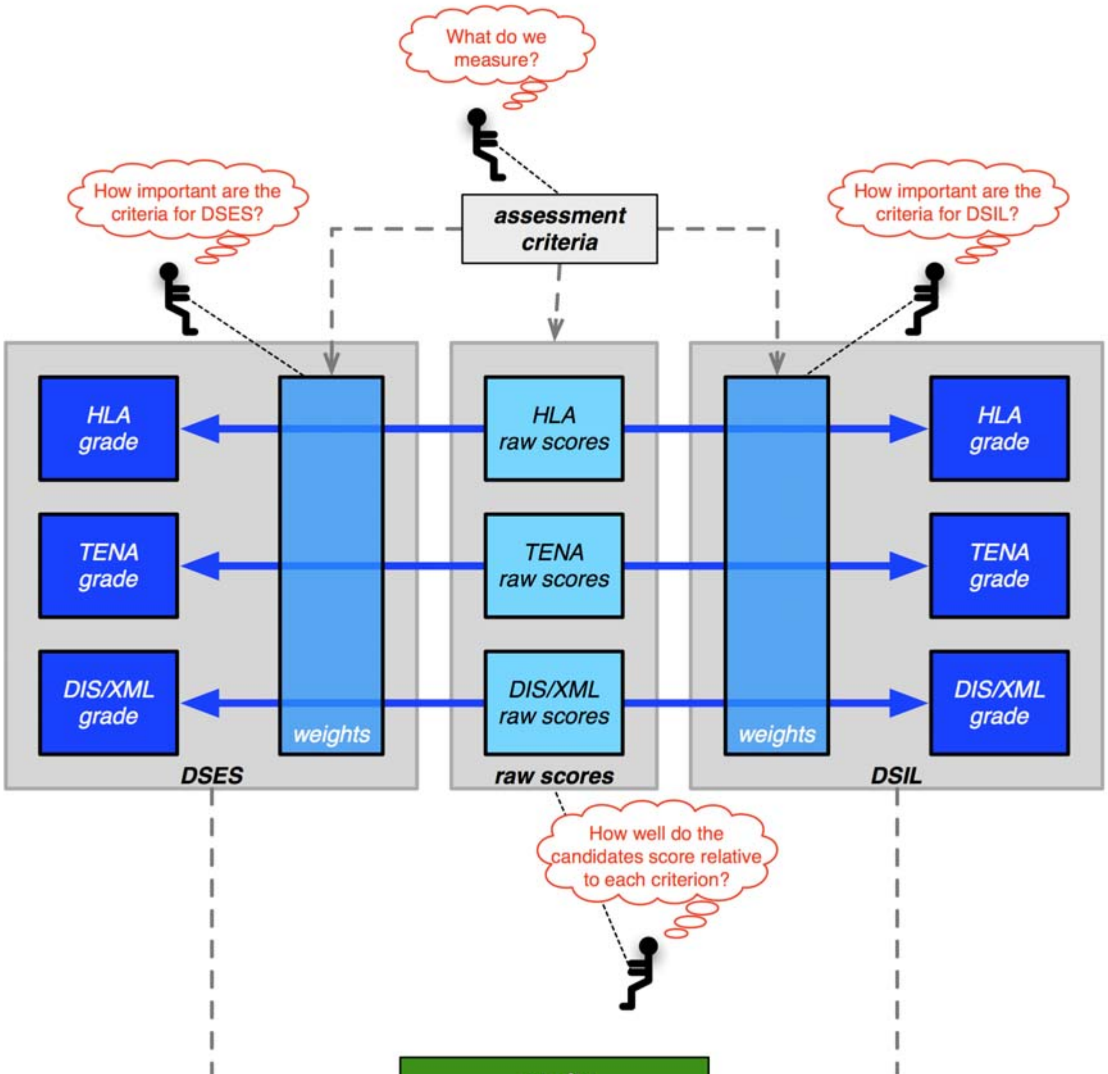
- **DSES**
- **DSIL**

our question

- assess the middleware candidates
- which is best suited to DSES & DSIL?

methodology

Analytic Hierarchical Process (AHP)



	HLA	TENA	DIS/ XML
DSES	?	?	?
DSIL	?	?	?

- background
- objectives
- methodology
- results
- conclusions

criteria

- **3 high-level categories**
  - operational factors
  - performance
  - programmatic factors

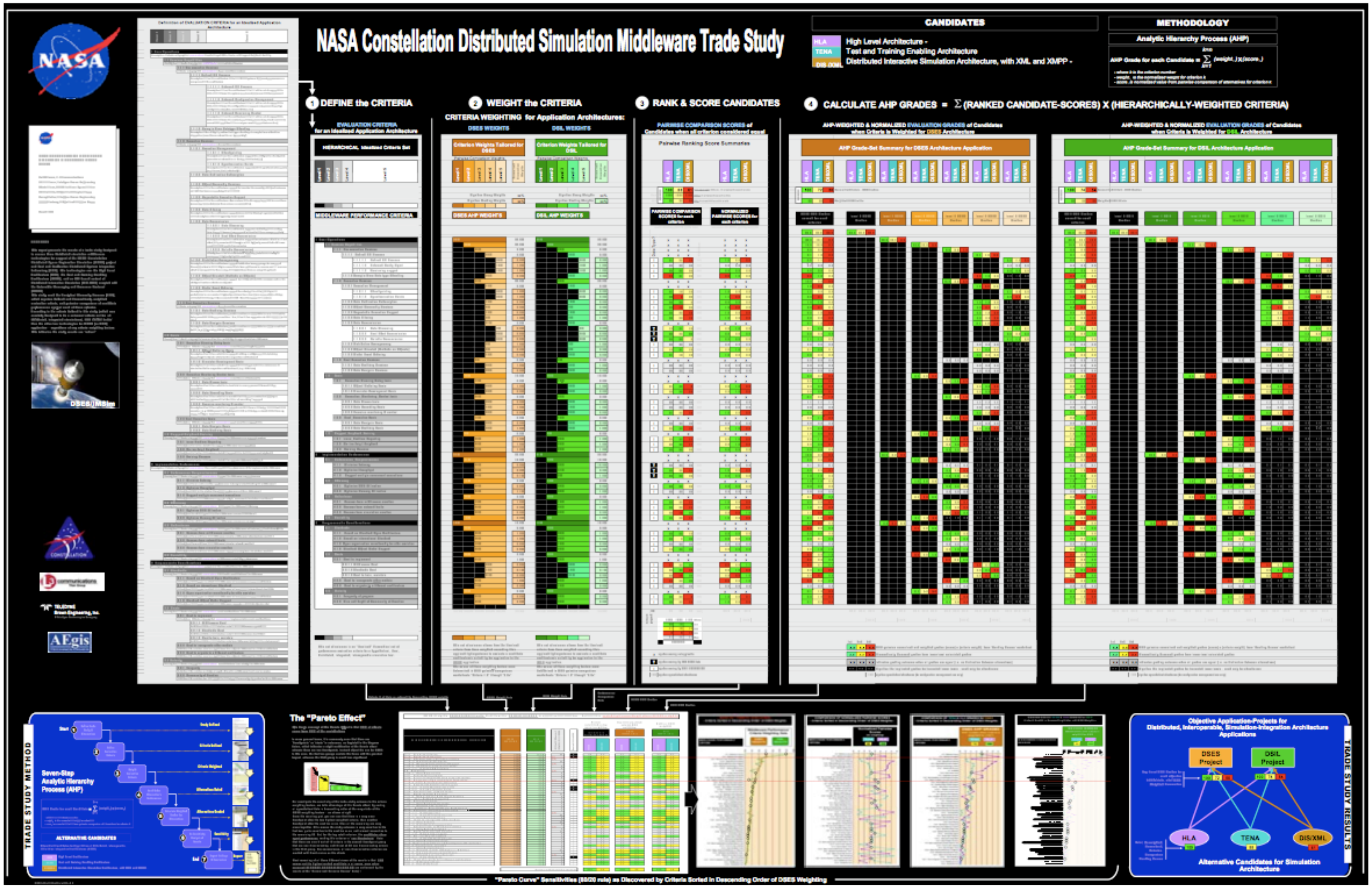
- **approximately 50 criteria**

- **examples**
  - ability to checkpoint
  - ability to synchronize simulations
  - latency & throughput
  - cost & training

weights

- **criteria not equally relevant to DSES & DSIL**
- **example:**
  - time management is not as important to DSIL as it is to DSES
- **AHP uses relative weights to determine overall grades**

results



- background
- objectives
- methodology
- **results**
- conclusions

overall grades

	HLA	TENA	DIS/XML
DSES	45	32	23
DSIL	44	32	24

conclusions

interpretation

- **results same for DSES and DSIL**
  - criteria do not differentiate between the two
- **DIS/XML clearly falls short**
- **HLA comes out ahead of TENA. Why?**

why?

- **which criteria are most significant? (Pareto effect)**

- **most significant differentiators:**
  - network throughput

- **other leading differentiators:**
  - crash robustness
  - global event ordering
  - repeatability
  - monitoring and control
  - software engineering process
  - scalability
  - longevity and depth
  - object model support

sensitivity

- **are the results sensitive to slight parameter variations?**

- **our analysis says "no"**

- background
- objectives
- methodology
- results
- conclusions

caveats

- **DSES and DSIL only (YMMV)**
- **criteria ok?**
- **weights ok?**
- **time-critical / high-frequency scenarios**