A TAXONOMY OF MULTIPLE FEDERATION EXECUTIONS

MICHAEL D. MYJAK
THE VIRTUAL WORKSHOP, TITUSVILLE FL

RUSSELL L. CARTER
CONCEPTUAL SYSTEMS AND SOFTWARE, CHINO VALLEY AZ

DOUGLAS D. WOOD AND MIKEL D. PETTY
INSTITUTE FOR SIMULATION AND TRAINING, ORLANDO FL

ABSTRACT

The High Level Architecture (HLA) supports the interoperation of sets of simulations within the context of a Federation Object Model (FOM), using the HLA Interface Specification services as provided by the Run-Time Infrastructure (RTI). Such simulations are federates and the set of federates is a federation. A run of a federation is a federation execution. Although the "normal" mode of operation is for a federate to operate in a single federation execution at any given point in time, the definition of HLA leaves open the possibility that a federate may be a member of multiple concurrently executing federation executions. In other words, two (or more) concurrent federation executions, of the same or different federations, could have one or more federates in common. Presumably the common federate(s) would exchange information between executions or otherwise use the events of one execution to influence another.

There are several distinct types of multi-federation executions. At the most basic level of classification, they can be broadly typed as either *bridged* or *hierarchical*. Bridged federation executions have one or more federates, called bridge federates, which are members of two (or more) federation executions. Recent literature has been primarily directed toward the common, or *bridge* federates which exchange (or transform) information between federation executions. In a hierarchical federation execution, one or more federates in the higher-level federation are composed of and implemented as lower-level federations, but appear as federates at the higher level. In this paper we develop a taxonomy of multiple federation executions, including examples.

AUTHORS' BIOGRAPHIES

Michael D. Myjak is Vice-President for Research and Development at The Virtual Workshop. Prior to that he was a Senior Research Computer Scientist at the Institute for Simulation and Training.

Russell L. Carter has a decade of experience in distributed computing, specializing in large-scale distributed applications and parallel computation. As Senior Scientist at Conceptual Systems & Software (www.consys.com) in Arizona, he is primarily responsible for development of advanced distributed object computing solutions. Mr. Carter's experience includes resource allocation issues in CORBA and HLA applications and computational mathematics analysis on high performance distributed computing architectures. He has authored several publications on distributed computing architectures, floating point arithmetic, performance analysis of distributed systems, and system software for distributed computing architectures.

Douglas D. Wood is a Research Computer Scientist at the Institute for Simulation and Training. Mr. Wood has performed research primarily in the area of distributed simulation, including HLA experiments, electronic warfare protocols, emergency management training, and algorithms for computer generated forces. Mr. Wood received a M.S. and a B.S. in Computer Science from the University of Central Florida.

Mikel D. Petty is an Assistant Director of the Institute for Simulation and Training. Dr. Petty leads IST research in distributed simulation, including High Level Architecture and Computer Generated Forces. Dr. Petty has worked in distributed simulation since 1990. He has received Ph.D., M.S., and B.S. degrees, all in Computer Science.

A TAXONOMY OF MULTIPLE FEDERATION EXECUTIONS

BACKGROUND

The High Level Architecture

The High Level Architecture (HLA) supports the interoperation of sets of simulations within the context of a Federation Object Model (FOM), using the HLA Interface Specification services as provided by the Run-Time Infrastructure (RTI). Such simulations are federates and the set of federates is a federation. A run of a federation is a federation execution. The FOM specifies a contract between members of a federation on the types of objects and interactions that will be supported across its multiple interoperating simulations. The HLA Runtime Infrastructure (RTI) provides the services defined by the HLA Interface Specification, through which the simulations communicate. The HLA Rules further establish the manner in which the federations and federates must comply with the OMT and RTI Interface Specification. It is within this framework that federates, within federations, interoperate (Figure 1).

Historical Comments

In recent years we have witnessed the evolution of *federated* simulation systems. Federated simulations claim to enhance the development, deployment and interoperability of *federates* (e.g., independently developed simulation applications, components or related tools) that build upon a rich heritage of simulator interoperability. We can trace the roots of modern federated simulations to the development of SIMNET in the early 1980's, but tracing the development of *interoperability* among distributed systems is a little less direct.

As SIMNET experience evolved, standardization with interoperability as a goal followed in the guise of Distributed Interactive Simulations (DIS) in the late 1980's and the Aggregate Level Simulation Protocol (ALSP) in the early 1990's. Many DIS and ALSP-based systems are still in use today. Integrating these two concepts together (DIS and ALSP) has been dubbed Advanced Distributed Simulation (ADS) and is seen as another development in the development of hybrid simulation systems. ADS concepts, upon which the High Level Architecture is based, are the legacy that grew out of SIMNET and distributed networked computing age, and is the basis for this paper.

Consider the evolutionary changes in software and networking during the evolution of SIMNET and ADS that permitted dissimilar simulation applications to interoperate and evolve. Recall that this was a time before the *Commodity Internet* (as we now know it today) existed. In 1984, the Defense Advanced Research Projects Agency (DARPA) released the management of the ARPANet over to the National Science Foundation declaring that Inter-Networking was no longer an advanced research application but was at this point in time an engineering project.

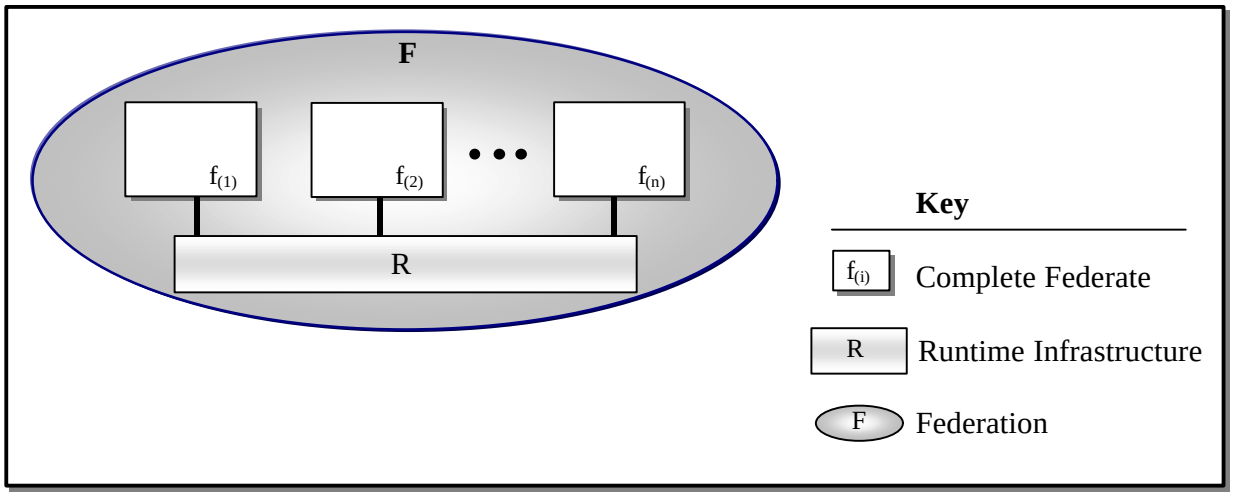

Figure 1. Typical HLA federation.

Nevertheless, when that transition to NSF occurred there were barely 100K nodes attached to the ARPANet, Intel 8086 computers were hot, and the Apple Macintosh had just been released. Bridges and Gateways on networks were the norm, and there were literally dozens of different, generally non-interoperable protocols available. The concept of general interoperability eluded us. So it is quite understandable that SIMNET was not based upon the Internet Protocol (IP) experience as the practice and use of IP was limited. People did not know the level of interoperability that would one day be achieved with IP. It would one day change the world.

In the 1970's and 1980's, many applications and products (e.g., simulation systems) were not interoperable. Quite the contrary, these were one-off solutions dedicated to specific platforms, operating systems, or vendor-specific implementations. Computing in this age was expensive. Software was rarely portable as porting costs were prohibitive expensive. Still, many users and developers alike were concerned about software reuse. Heterogeneous environments were slowly to evolving, but interoperability in the general sense was not achievable. For these reasons and more, the International Standards Organization (ISO) developed the Open Systems Interconnection (OSI) initiative.

The acceptance and popularity of IP continued to grow, as did interoperable systems based on the OSI model. With the introduction of the NSFNet, routing became the norm and the dominance of IP became clear. The number of "other" protocols began to diminish rapidly. However, the simulation community was slow to adapt to this new infrastructure model as neither ALSP nor DIS version 1.0 were based on the OSI protocol stack. In 1991, the NSF lifted its restriction on commercial use of the NSFNet and the modern Commodity Internet was born. Along with birth of the Internet came the first "point-n-click" Browser, called *Gopher*. Two years later we saw the evolution of *Mosaic* and the birth of the World Wide Web in 1993. Thus began one of the greatest success stories for interoperability as the Internet was then growing at an astounding rate of 341,634%. By 1996 the Internet supported over 150 countries and connected over 10 million hosts. Consequently, the next version of DIS aligned itself with the Internet Protocol Suite.

This is the success story of interoperability in heterogeneous environments. Today the value of a standards driven, system-independent communications mechanism is well understood. Software development costs are controlled because of vendor competition and platform independence. Therefore the long term success of the HLA and interoperable federations will hinge on the ability to standardize HLA in terms of OSI.

Relatively speaking, distributed simulation is a small community (when compared to the World Wide Web). But there are some significant changes on the horizon. Applications such as distributed simulation and shared immersive environments had been presented as the driving force behind the development of the Next Generation Internet. As an application group, this type of distributed application is quite capable of stressing the low-latency, high-bandwidth capabilities of even the fastest network infrastructure. Furthermore, the application and middleware layer technology being developed today in the simulation community may one day find its way into a wide variety of non-military applications, such as immersive telemedicine, teleinstrumentation, and entertainment. So the technology, properly applied, holds great promise.

A middleware layer like the High Level Architecture (HLA) is considered to be a relative newcomer to distributed computing field. While well recognized as both feasible and useful, HLA is a middleware product that does not fit neatly into any one layer in the current OSI 7-layer model, thus interoperability concerns exist. Today HLA is forging new ground and is the basis behind the development of *federated* simulation systems. Sponsored and developed by the Defense Modeling and Simulation Office (DMSO), the HLA provides both an architectural *framework* and interaction specifications that address the interoperation of dissimilar simulations at the Application Program Interface (API). HLA is currently being investigated for use in acquisition confirmation, software integration and testing, as well as the reuse of legacy simulation systems.

Federates and Federations

As stated in the HLA Interface Specifications (DoD, 1998b), "A *federation* is the combination of a particular FOM, a particular set of federates, and the RTI services." This model of interoperation is generally considered to be self-contained. That is to say that the basis for

communication, as described by the FOM, occurs among federates during a federation execution. However, it is conceivable (and perhaps inevitable) that simulations may interoperate across FOMs, and in a multiple federation configurations. This notion of multiple federation interoperation and how such federation configurations might be structured requires careful interpretation of the HLA rules (DoD, 1998a), as will be discussed later. The rules are given here for reference (Figure 2).

The original developers of the HLA had intended that a simple, single federation model (e.g., federates, FOM, and RTI) would suffice for the vast majority of federated simulations. Indeed, some considered this architecture to be complete. However, some other federation models were left open in the definition of HLA, as we shall see.

INTRODUCTION TO MULTIPLE FEDERATIONS

Definition

We define a *multiple federation* (or *multi-federation*) as a set of more than one currently executing federations to which one or more federates are simultaneously joined. Presumably the common federate(s) would exchange information between executions or otherwise use the events of one execution to influence another.

The Precedent For Multiple Federations

The HLA proto-federation experiments were quickly followed by a proposed HLA security architecture design (Filsinger, 1997). That architecture indicated that a single (i.e., common) FOM would not suffice for federation executions requiring multiple security levels. Such federations would require some special action to be taken in handling classified data (e.g., secret-high to unclassified). To resolve these issues, the HLA security architecture proposed the use of a security guard process. The guard process would "scrub" the classified information from data flowing to unclassified federates and/or augment the unclassified data with relevant classified information when flowing in the reverse direction. The security details, while interesting, are not of concern in this paper.

However, what is of interest here is that the security architecture design clearly is based on a single federate (the "HLA Security Guard Federate") connected to two concurrently executing federations (Filsinger, 1997). This architecture provides the precedent and initial example of multi-federations.

Federation Rules	
Rule 1	Federations shall have an HLA federation object model (FOM), documented in accordance with the HLA object model template (OMT).
Rule 2	In a federation, all simulation-associated object instance representation shall be in the federates, not in the run-time infrastructure (RTI).
Rule 3	During a federation execution, all exchange of FOM data among federates shall occur via the RTI.
Rule 4	During a federation execution, federates shall interact with the RTI in accordance with the HLA interface specification.
Rule 5	During a federation execution, an attribute of an instance of an object shall be owned by at most one federate at any time.
Federate Rules	
Rule 6	Federates shall have an HLA simulation object model (SOM), documented in accordance with the HLA OMT.
Rule 7	Federates shall be able to update and/or reflect any attributes of objects in their SOMs and send and/or receive SOM interactions externally, as specified in their SOMs.
Rule 8	Federates shall be able to transfer and/or accept ownership of attributes dynamically during a federation execution, as specified in their SOMs.
Rule 9	Federates shall be able to vary the conditions (e.g., thresholds) under which they provide updates of attributes of objects, as specified in their SOMs.
Rule 10	Federates shall be able to manage local time in a way that will allow them to coordinate data exchange with other members of a federation.

Figure 2. HLA Rules.

Therefore, the question is no longer one of whether or not such multi-federations *can* interact, but rather *how* will two (or more) federations interoperate. Since HLA provides no defined support services for multiple interoperation, a secondary question must be raised as to what kind of additional services might be required to facilitate such interoperability.

Similar issues have recently been raised relative to other scenarios (e.g., multiple levels of detail, differing fidelity requirements, etc.) all of which appear to require or at least involve multi-federation configurations.

Multiple Federations And The HLA Rules

On the surface, it would appear by definition that multiple concurrently executing federation executions can not interact with one another. After all, a federation is defined by its federates, a singular FOM, and an RTI. However, non-interoperable federations can diminish the overall potential (not to mention the scalability) of federated simulation technology. Multi-federations are only interesting when inter-federation communications can exist. But how is this to be accomplished? That involves a thorough understanding of the HLA rules.

The HLA Rules describe precisely how federates and federations are permitted to interact (Figure 2). Note that these rules state specifically how and under what conditions the RTI Application Program Interface and Federation Object Model are to be used within a federation. We note that according to the draft rules, multi-federation interaction is tenuous at best. One interpretation of the rules indicates that all federations must take into account all of the data being represented by all relevant FOMs; this as some have suggested, is called the "SuperFOM" model. An alternative to the SuperFOM approach would be to utilize mechanisms and methods of communication *outside* of the federation, to communicate with other entities (e.g., federates, sub-components) "outside" of the RTI. This "out of band" communication mechanism can be used to communicate FOM data outside of a given federation if it is not directed at another federate *within* the source federation.

TYPES OF MULTIPLE FEDERATIONS

Bridged Federations

The HLA rules do not specifically prohibit or permit a federate from being attached to multiple, concurrent federations. Indeed, this leaves the whole subject of multi-federation interoperation in a rather gray area. HLA federate rules indicate that the federate must have a SOM that specifies [completely] the elements *and* interoperability requirements (e.g., classes, attributes, interactions, ownership, etc.) of that federate. The current draft of the HLA federation rules appears to indicate that only one FOM is permitted in a federation, and that "… all exchange of FOM data among federates shall occur via the RTI." (Note the singular reference to both RTI and FOM.)

This would imply that bridge federates which support an additional interface for FOM Data (Figure 3) are not directly permitted.

Definition. So if multiple federate interoperation is not directly permitted, and direct federate attachment to multiple RTIs for the purpose of passing FOM data (outside the RTI) is directly prohibited, how are multi-federations to interoperate? The answer lies with the design of the guard federate mentioned above. If we except this solution as the prototypical example, requiring the mix of dissimilar FOMs into a single "federation execution," then the mechanisms employed by the guard federate should be investigated.

In principle, the bridge federate as guard (shown in Figure 3), permits FOM data to flow from federation $\mathbf{F_1}$ into federation $\mathbf{F_2}$ (and visa versa), like a bridge (hence, *bridge federate*). Bridge federates perform various scenario-dependent functions such as performing data replacement, data composition, data decomposition, or other data transformation operations. Recent experiments of this type, described in (Beebe, 1997) and (Bouwens, 1998), have been characterized by the configuration of one federate as a member of at least two federations, even though this type of connectivity is not explicitly permitted by Draft HLA Rules. Instead, (Bouwens, 1998) indicated that in this configuration, the bridge federate is actually more complex, in that a separate federate ambassador and FOM is used for each adjoined federation.

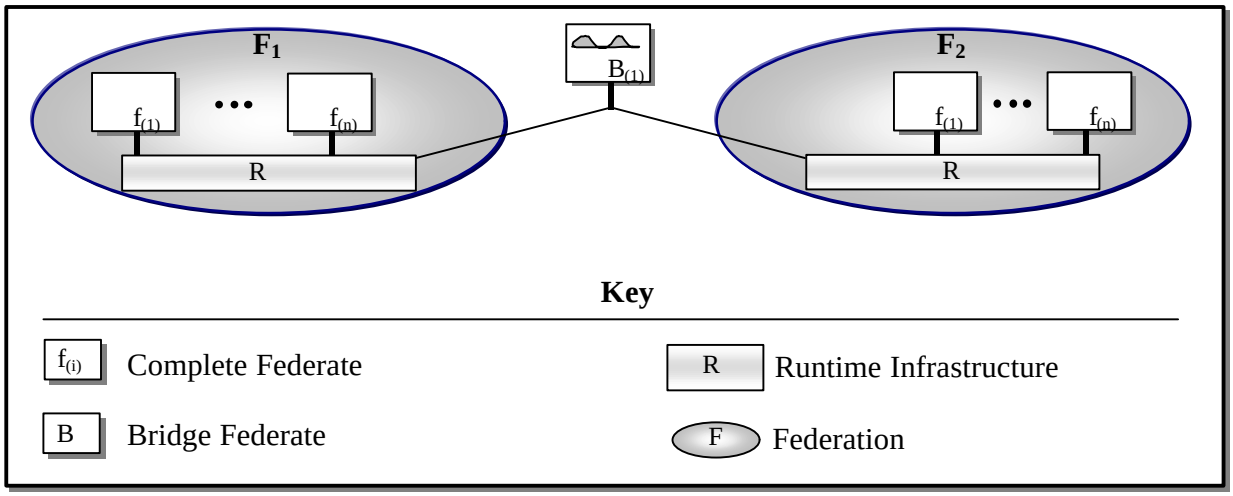

Figure 3. Typical acclaimed bridge federate configuration.

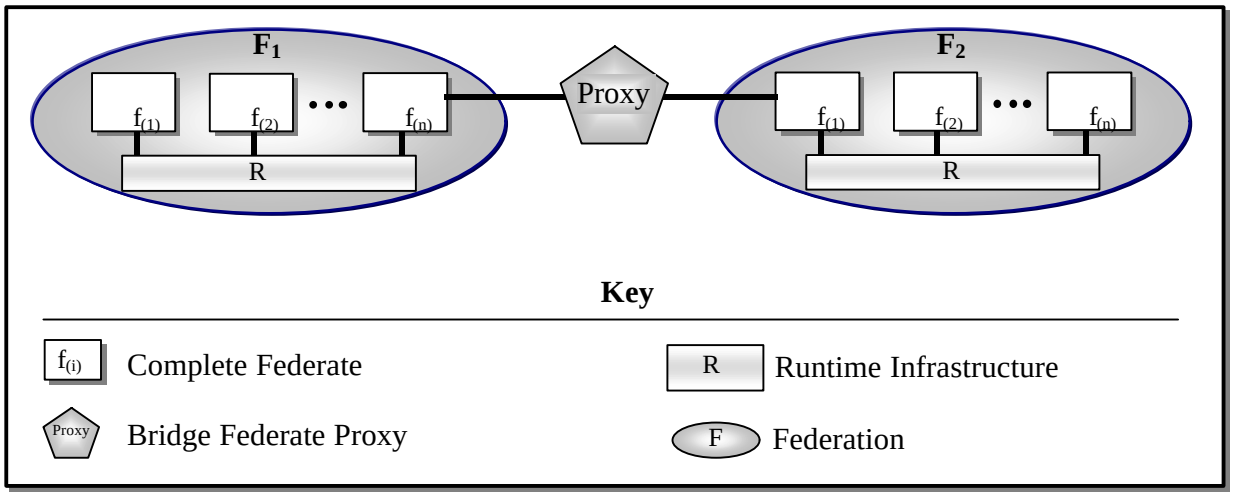

Figure 4. A typical proxy federate architecture.

In addition, a transformation function is required to map the FOM data from source federation into representative FOM data for the destination federation. Since these transformation functions are not unidirectional, bi-lateral transformation functionality must be contained within the federate. Therefore, since the FOM data is actually being modified, HLA Rules #1 and #3 have not been broken; the exchange of FOM data does occur *through* the RTI.

Examples. Therefore, in order for bridge federates to be able to exchange FOM data, they must be more complex, supporting a distinct federate and RTI interface for each federation for which they are joined, plus they must act as proxies handling the additional transformation function(s) (see Figure 4). In short, the proxy must manage the flow of information from federation (**F_1**) to federation (**F_2**) by performing data replacement, data composition, data decomposition or some other form of data transformation (and visa versa, from federation (**F_2**) to federation **(F_1**))

We can now characterize a bridge federate as a proxy service, managing asymmetric data flows, and applying bi-directional transformation functions as necessary, dependent upon the FOM data for the two federations being adjoined. Given this definition of a bridge federate, it should be clear that a universal bridge federate will not be possible. Each bridge federate must build a unique set of transformation operations between

any two representative FOMs. And since the transposition of such data is not reciprocal, each data flow will require its own transformation matrix.

Hierarchical Federations

While security guard federates have received the most attention of late, they are by no means the only example or possible interpretation for handling multi-federation interaction. As proposed, bridge federate performs a rather complex set of functionality (i.e., asymmetrical, bi-directional data transformation between two federations). In this regard, they are essentially providing a proxy service to their adjoined federates. Their differential information transformation between the destination federation on behalf of the federates in the source federation makes them appear (in the destination federation) as locally attached federates.

Definition. It is the generic concept of a federate acting as a proxy for another federate (Figure 5) that can be extended to support hierarchical federation executions. For a proxy federate need not just represent the interactions of one (or more) federates from one federation into another. A proxy could, for example, represent an entire federation as being a single federate (Figure 6). Federations built using such proxy services are hierarchical federations. In theory, a multiple-level hierarchy of federations would be possible. Furthermore, because of the bi-directional transformation functions supported by a proxy federate, they are able to bridge or *gateway* non-HLA simulation applications to represent complete federates or federations. A proxy federate may therefore represent an entire DIS simulation, or an ALSP confederation within an HLA federation. For example, the DIS Gateway federate based on the RPR FOM and developed by the Institute for Simulation for Training (IST) is one such representation (Wood, 1997). Thus the role of a proxy acting on behal*f of* another federate, or federation, or generic simulation application within another federation, represents an untapped, future potential in the overall HLA design.

Examples. Federate simulations are assembled into federations using federates (often independently developed simulation applications) as the essential building blocks. Under HLA, a typical federation (**F**) is composed of a collection of federates ($\mathbf{f}_{(i)}$), interoperating with one another through a common Run Time Infrastructure (**R**). The Federation Object Model (FOM) defines or identifies the information communicated between the various federates. A typical HLA federation might, for example, be composed of 3 or more federates sharing a single FOM and common RTI.

Under HLA, a common method of invocation is to associate a simple federate process with a unique host or host process (i.e., thread), and coupling that federate to the HLA federation by way of a singly attached interface (e.g., the federation's RTI). A component federate however, can be described as being a more complex federate, often composed of several parts (see Figure 7). In this taxonomy, each sub-federate ($\mathbf{Fs}_{(I)}$) component is typically associated with one host or one host process.

Component federates are said to be complete when the federate they compose is represented in its entirety, by all sub-federate processes. When taken together, the components provide a unique and complete representation (i.e., a mutually exclusive interface) into the RTI (Figure 8). In other words, the sum interface product of all sub-federate components yields a complete federate API to the federation's RTI. A component federate is only considered "complete" when all essential state data (i.e., object classes, attributes, and interactions) are all uniquely represented by the necessary and operational components, and when these components are connected to the RTI. Component federates may however, be out of scope given the draft HLA rules. There is some question as whether rules 3, 4, and 5 apply to components of a federate or entire federates.

A typical federation comprised of component federates may actually appear to the RTI as several incomplete federates, rather than connected components of a single federate application process (Figure 5). Also since FOM data may be passed between components outside of the RTI (such as in Figure 9), it is unclear to the author whether this also violates HLA rules.

ISSUES

It should make little architectural difference whether or not a federate application is managed by a single processor, is supported by multiple process threads, or is distributed across several platforms. Within a single host, Intra-federate communications occur outside of the HLA RTI. So whether a federate is composed of components that reside on separate hosts, or run as multiple threaded processes within a single host, should

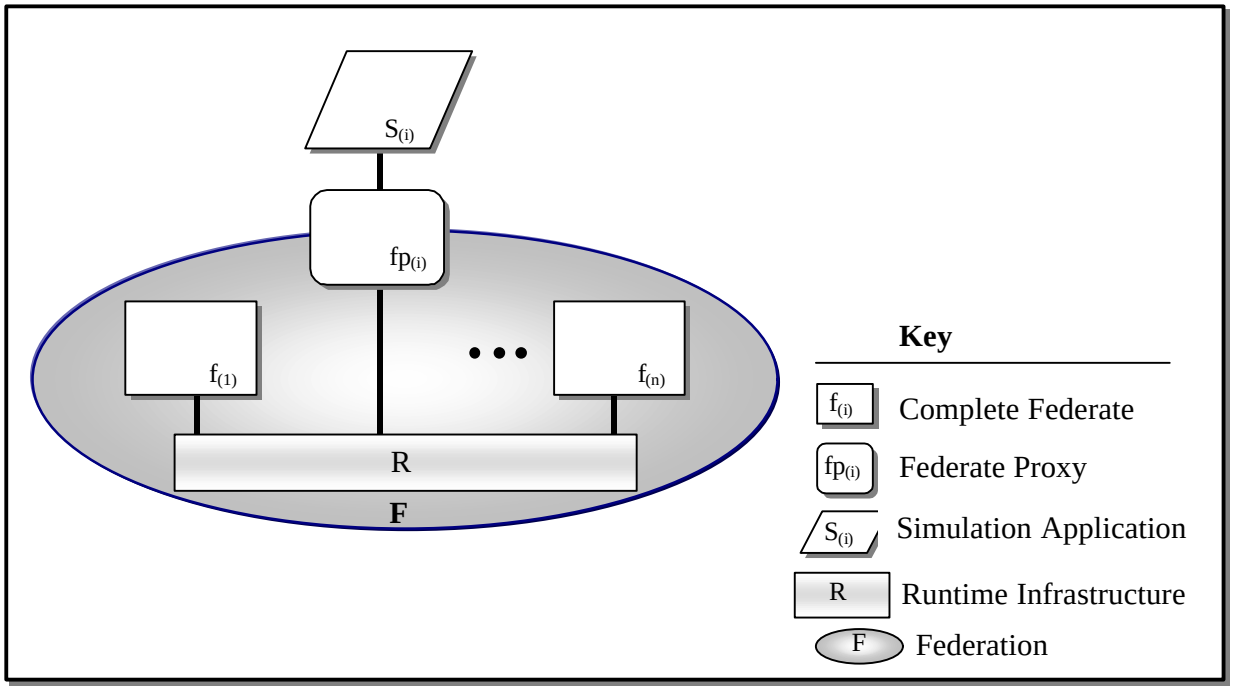

Figure 5. Component federation with basic proxy.

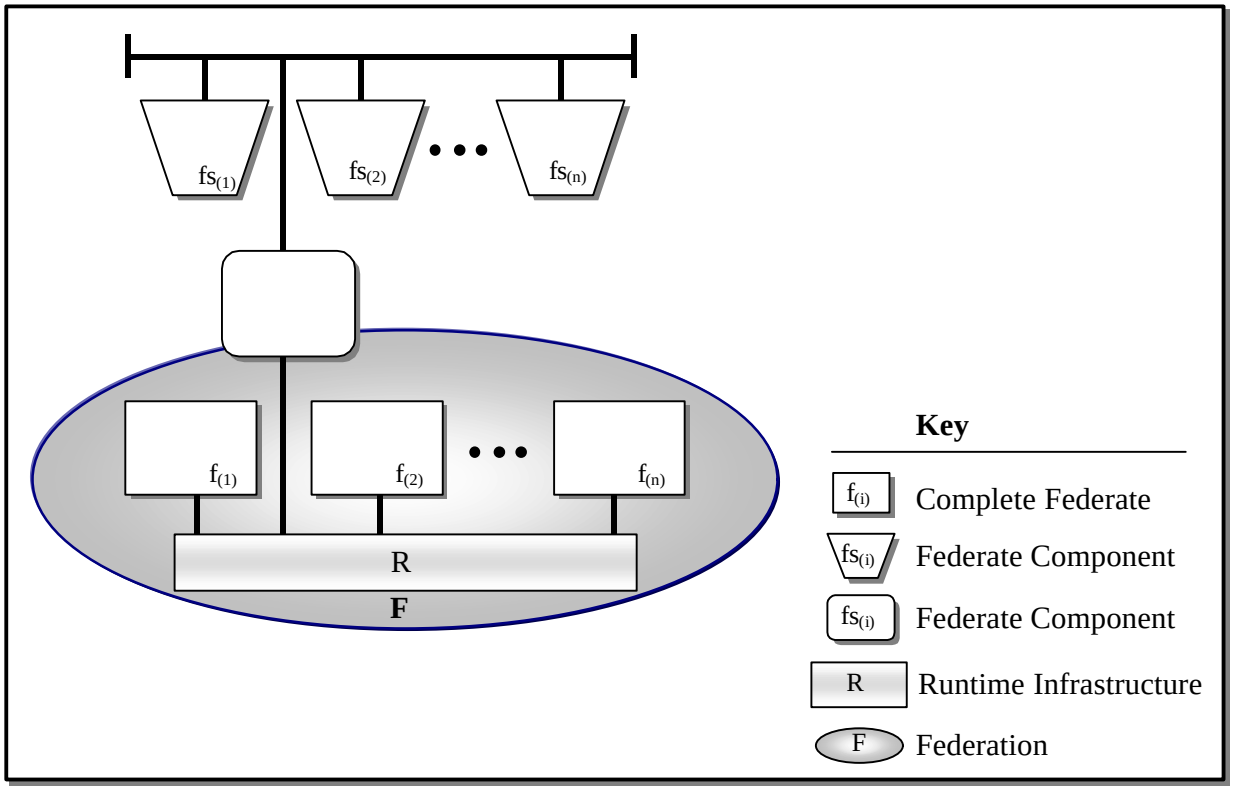

Figure 6. Federation proxy as a component federate, representing a non-HLA federation.

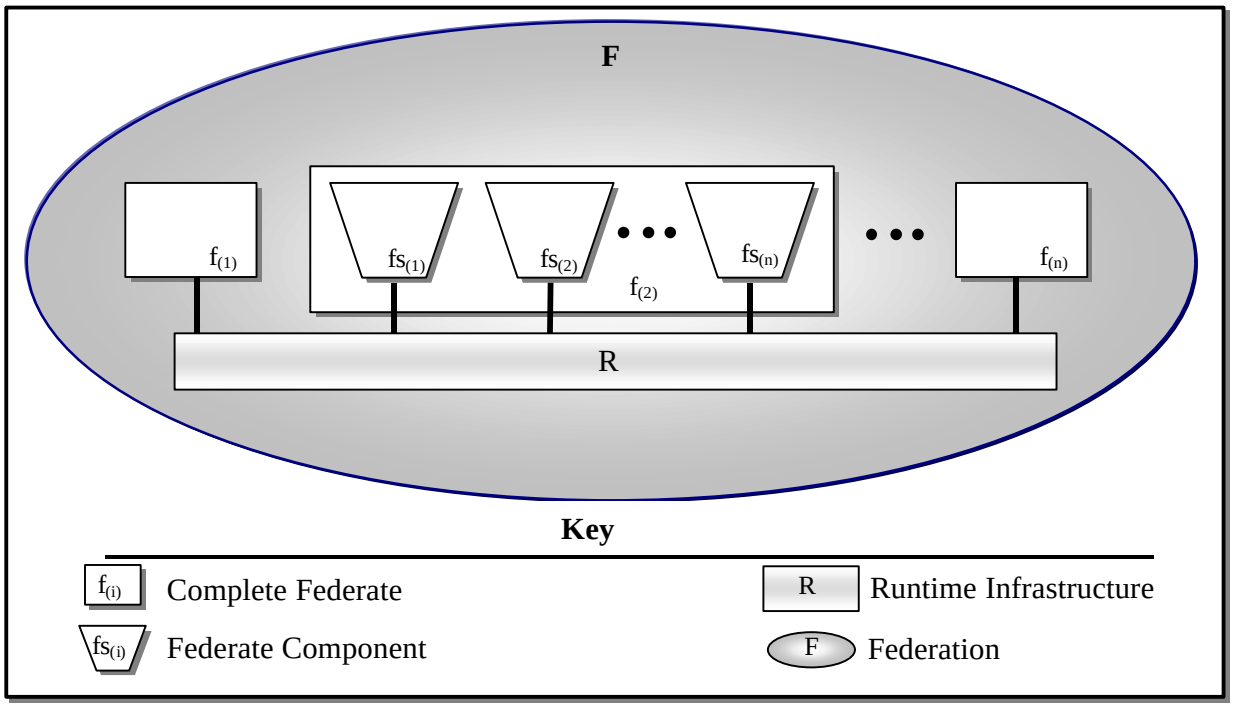

Figure 7. HLA federation containing a component federate.

pose little or no difference to a conformant HLA federation. The end result is that the supporting architecture should be modular and robust enough to support these variants.

Assuming a modular distributed architecture, a component federate implementation may represent a significant performance improvement for certain applications (e.g., real-time simulations). This improvement may be obtained by bringing more processors to bear on the computationally intensive aspects of the simulation, without unduly increasing communications load through the RTI.

Many types of complex high fidelity simulations are naturally distributed and require moderate bandwidth exchange of interface data. An example is the multidisciplinary coupling of elastic structural and fluid dynamics solvers in the modeling of aircraft flight dynamics. Whereas the other federates in the federation at large may be interested only in location, displacement, and other external characteristics of the complete federate simulation object(s), high fidelity often requires significant exchange of internal data among the distributed federate components. Thus, by additionally partitioning the network load by federate component functionality (rather than solely by simulation object locality), the scalability of the overall HLA simulation is enhanced. This functional partitioning is explicitly defined in a hierarchical multi-federation architecture.

In the single platform configuration, each component process (i.e., federate) operates independently (as in Figure 7). When component federates are partitioned over multiple hosts, the sub-federate applications must be partitioned intelligently, such that they each provide a unique (for the federate) object, attribute and/or interaction resolution with the RTI. (Current RTI services do not distinguish between multiple processes when identifying a federate instance. RTI services will need to be enhanced in order to fully exploit this architectural advantage.) Currently, there is no support through HLA to permit intra-federate communications of FOM or SOM data to occur, even if the federate designer has developed multiple, yet mutually exclusive interfaces to the RTI for each of the component federate process (Figure 7 and Figure 8).

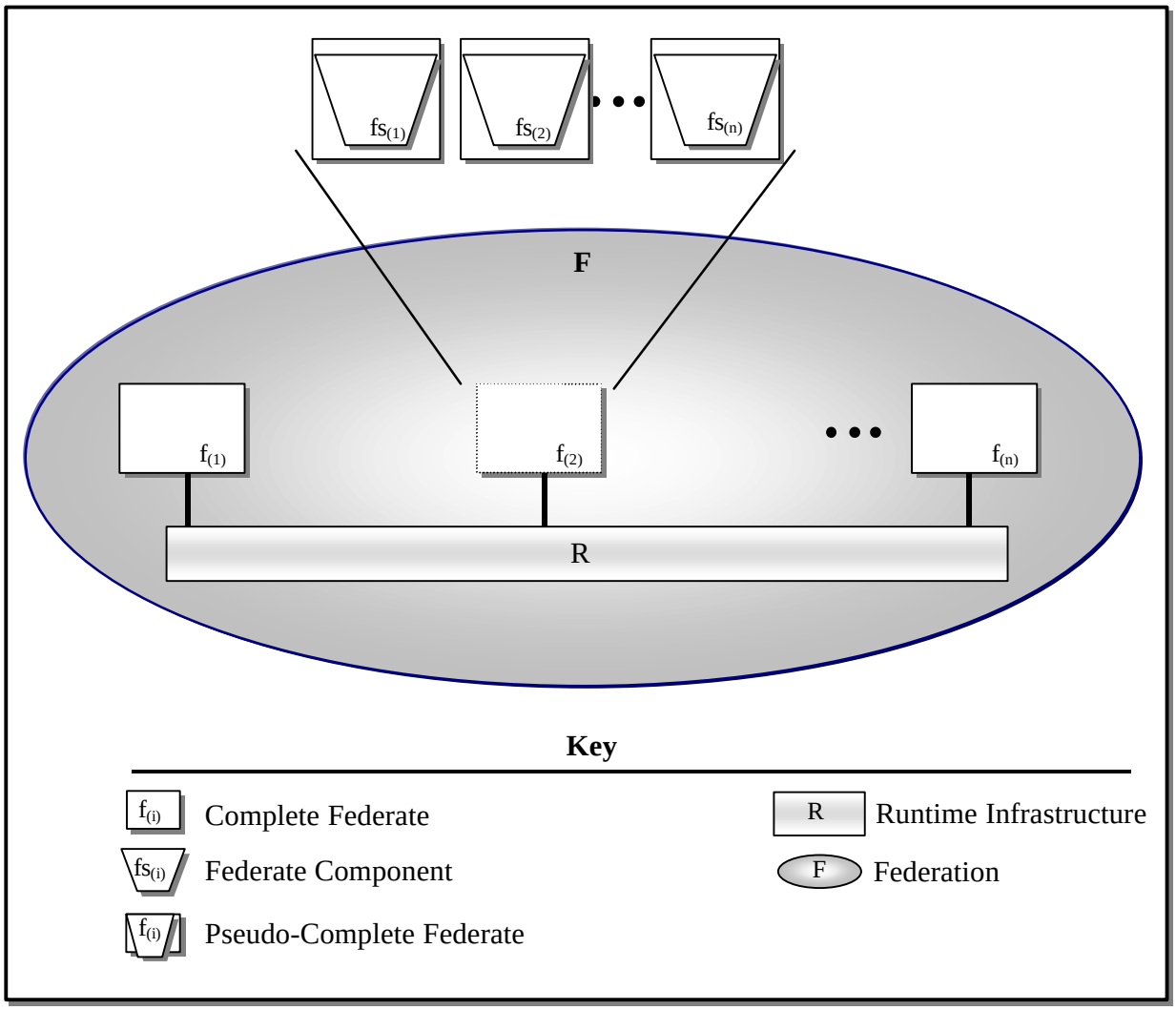

Figure 8. Logical manifestation of a component federate, implemented over separate hosts.

SUMMARY

Multi-federations exist where one or more federates are members of at least two concurrently executing federations. Such federates exchange information between the federations. A federation where a federate is used to bridge between classified and unclassified federation executions is the prototypical multi-federation.

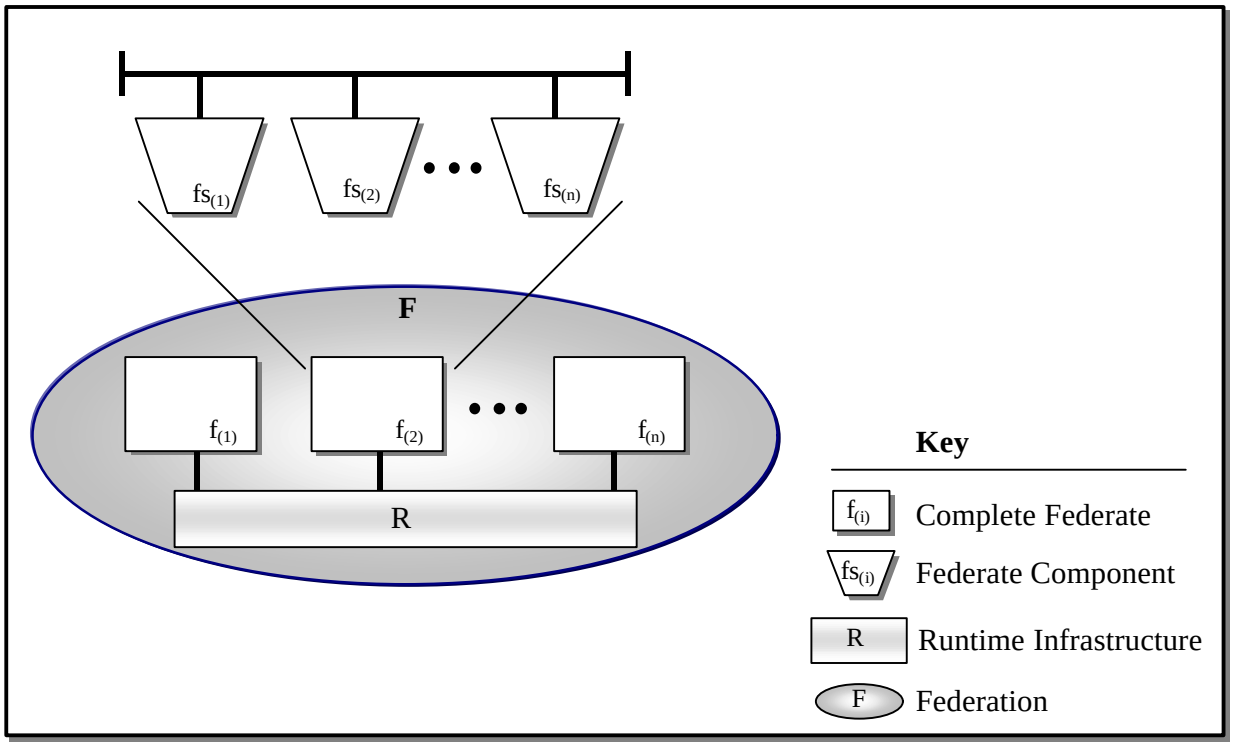

Figure 9. Component federates as part of the federation.

REFERENCES

Beebe, B., Bouwens, C., Braudaway, W., Harkrider, S., Ogren, J., Paterson, D., Richardson, R., and Zimmerman, P., (1997). "Building HLA Interfaces for FOM Flexibility: Five Case Studies", *Proceedings of the 1997 Fall Simulation Interoperability Workshop*, Orlando FL, September 8-12 1997, pp. 1027-1034.

Bouwens, C., Hurrell, D., and Shen, D. (1998). "Implementing Ownershop Management with a Bridge Federate", Proceedings of the 1998 Spring Simulation Interoperability Workshop, Orlando FL, March 9-13 1998, pp. 1126-1131.

DoD (1998a). "Draft Standard [For] Modeling and Simulation, High Level Architecture (HLA) – Framework and Rules", U. S. Department of Defense, Document available on-line at http://hla.dmso.mil, February 5 1998.

DoD (1998b). "Draft Standard [For] Modeling and Simulation, High Level Architecture (HLA) – Federate Interface Specification", U. S. Department of Defense, Document available on-line at http://hla.dmso.mil, February 5 1998.

Filsinger, J., (1997). "HLA Security Guard Federate", *Proceedings of the 1997 Spring Simulation Interoperability Workshop*, Orlando FL, March 3-7 1997, pp. 1015-1023.

Wood, D., Cox, A., and Petty, M. (1997). "An HLA Gateway for DIS Applications", *Proceedings of the 19th Interservice/Industry Training, Simulation, and Education Conference*, Orlando FL, December 1-4 1997, pp. 511-520.