Standardising Distributed Simulations: The High Level Architecture

Marco Brassé and Nico Kuijpers

Computer simulations have many application areas ranging from physics and chemistry research, biomedical engineering and weather forecast to military training and the preparation for missions in hazardous environments. Distributed simulations that involve humans-in-the-loop are typically applied for training and preparation of large, often international, military operations. The need for standardisation is clear from this observation. No wonder the military world, especially the United States Department of Defence, is the driving force behind the development of standards and technologies for distributed simulations, which resulted in the High Level Architecture. HLA is widely applied and is in the process of becoming an IEEE standard.

Distributed Simulation

Nowadays, the demand for distributed simulation is increasing for several reasons: budgets for large-scale training operations are decreasing and the need for being well-prepared to participate in a large, often international, operational team is increasing. Simulations are especially applied where people train for hazardous situations like warfare or prepare for space missions. Large-scale distributed simulations are of special interest to decrease the costs and organisational effort of, for instance, disaster preparation, through replacing humans that play a minor role by computer programs. Thus, application areas for distributed simulations are not restricted to the military domain, but can also be found in the civil and space domains.

As technology improves and the costs of hardware decreases, large-scale distributed simulations are becoming affordable, especially when compared to real-life exercises. Distributed simulations require real-time data exchange between simulators (such as fighter aircraft simulators) that are already complex themselves. This implies the demand for an interoperability standard which not only foresees in communication standards but also in a generic (software) development process. Since large pieces of complex, and thus expensive, software are involved, there is also a demand for re-using components. Finally, legacy systems, in most cases simulators that are being used to operate stand-alone, need to be taken into account.

Simulation Interoperability

Distributed simulations can be categorised as real-time simulations and event-driven simulations. Real-time simulations are mostly applied for training and mission rehearsal, whereas event-driven simulations are applied for operational analysis and operations research. Both types of simulations have their own background and standards in the military world. The US DoD developed the DIS (Distributed Interactive Simulation) protocol to enable the linking of real-time training simulators and ALSP (Aggregate Level Simulation Protocol) for event-driven simulations [1, 2]. In 1995 the US DoD initiated the development of a generic framework for distributed simulations, called the *High Level Architecture*.

Distributed Interactive Simulation

DIS (Distributed Interactive Simulation) is a standardised protocol for interconnecting large numbers of heterogeneous simulators across local and wide area networks. DIS allows the combination of such diverse elements as real-time interactive human-in-the-loop simulators, computer generated autonomous agents, numerical simulations of physical processes, and live instrumented participants, to operate together in one shared synthetic environment.

DIS has been widely accepted, both by industrial and governmental institutes. The technical approach of DIS is based on the following key elements:

- *Object/event architecture.* Participating simulation objects are called *entities.* Dynamic entities inform all simulations of their status and actions through the transmission of standardised information packets, so-called Protocol Data Units (PDUs).

- *Autonomy of simulation nodes.* All events caused by an individual simulation are broadcast to all other simulations. The receiving simulations are responsible for determining whether an event is relevant, and for calculating the effects caused by the event. This principle enables individual simulators to enter and leave a DIS session without disruption.

- *Transmission of state change information only.* In order to minimise network traffic and processing time, simulators only transmit state changes of the entity they simulate. Continuous activities are transmitted at a reduced update rate. Each simulation extrapolates the states last reported by other simulations until the next state update is received. All simulations are required to transmit new state information when the discrepancy between their true state and the extrapolated approximations made by other simulations is becoming too large. This method is called *deadreckoning.*

DIS has been successfully applied in large military exercises with hundreds of simulators connected to each other. The deadreckoning mechanism preserves the network from being overloaded, although all information is broadcast to all participants. As a consequence of using UDP/IP for data broadcast, the protocol is not reliable, i.e., delivery of a message is not guaranteed. Another drawback is the rigidness of the protocol: new PDUs were being defined for new simulation needs. The call for reliability, flexibility and re-usability resulted in a new approach: the High Level Architecture.

High Level Architecture

In 1995 the US Defence Modeling and Simulation Office (DMSO) initiated a new development to establish standards for Modeling & Simulation, called the High Level Architecture. The HLA standard promotes re-use of simulations and their components. It attempts to specify the general structure of the interfaces between simulations without making specific demands on the implementation of each simulation. The standard is developed in a co-operative, consensus-based forum of developers [3].

HLA Terminology

A number of *federations*, each concentrating on a specific area of simulation, are envisaged. A federation consists of simulation applications called *federates.* Federates may be simulation models, data collectors, simulators, computer generated forces or passive viewers. A simulation session, in which a number of federates participate, is called an *execution.* Simulated entities are called HLA *objects.* All possible interactions between the federates of a federation are defined by the *Federation Object Model* (FOM). The capabilities of a federate are defined by the *Simulation Object Model* (SOM). The SOM is introduced to encourage re-use of simulation models. The FOM and SOMs may be regarded as contracts that function as interface specifications for the federate developers.

The state of each object is defined by its *attributes*. Attribute values can be passed from one object to another. Objects interact with each other via *interactions* which may be viewed as unique events. Initially, an object's attribute is controlled by the federate that instantiated the object. However, attribute ownership may change during the coarse of the simulation. Handing over attributes allows, for instance, the improvement of accuracy by changing to a more accurate simulation model.

In order to reduce network traffic and to limit the amount of computation each federate has to perform, HLA provides a mechanism of *publication* and *subscription*. Upon initialisation each federate registers object classes and associated attributes it will represent (publication). It also registers the object classes, attributes and interactions it needs to perform its task (subscription). Note that a federate can not only subscribe to attribute types, but also to (ranges of) attribute values. The aim here is to filter as much information as possible at the source. Publications and subscriptions are dynamic and may be changed during a session.

The HLA standard is formally defined by three basic concepts:

1. The *Interface Specification* is a formal, functional description of the interface between the HLA application and the underlying *Run-Time Infrastructure* [4].

2. A set of *HLA Rules* are defined to which HLA applications have to comply [5].

3. The *Object Model Templates* define the structure of the FOMs and the SOMs [6].

The *Run-Time Infrastructure* (RTI) is the implementation of the HLA Interface Specification and forms the basis for all HLA federates. The software layer takes care of communication between the simulation models. The RTI supports HLA federations through management services, divided in six categories [7]:

1. *Federation Management.* Create or join execution of a federation; for example, if a federate is the first member of a federation to begin execution, that federate can call the RTI to initiate execution of the federation. Once a federation execution has begun, a federate may call the RTI to join the federation.

2. *Declaration Management.* Establish data requirements; for example, a federate which has joined a federation will publish object attributes and interactions to the federation and subscribe to object attributes and interactions it wishes to receive from other federates.

3. *Object Management.* Create, register, discover and delete objects; for example, federates will register and update attributes of objects they have published to the federation, and they will discover objects generated by other federates.

4. *Ownership Management.* Transfer ownership of objects and attributes to or from other federates; for example, a federate currently responsible for updating an object may want to pass that responsibility to another federate. Also, a federate may acquire ownership of an object from another federate.

5. *Time Management.* Handling of messages in different ways, depending on the requirements of their destinations, e.g., in receive order (appropriate for real-time human-in-the-loop simulators), in priority order, in causal order, or in timestamp order. These last three methods are typically suited for non real-time event driven simulations for, e.g., analysis purposes.

6. *Data Distribution Management.* Can be regarded as an extension of Declaration Management by providing a mechanism to filter data based on the actual *values* of the attributes.

Using the Data Distribution Management (DDM) services, attributes can be associated with 'volumes', termed *regions*, in a user-defined multi-dimensional co-ordinate system, termed *routing space*. A published attribute can be associated with a so-called *update region* whereas an attribute that is subscribed to can be associated with a *subscription region*. In case a subscription region overlaps an update region, the (value of the) attribute will be communicated [8].

The DDM filtering mechanism should be used by defining the regions in terms of attribute values. This approach has been chosen in HLA to allow for a high degree of separation between the necessary (machine dependent) representation of attribute values on the one hand and the issue of communicating an attribute value on the other hand. It is stressed that the HLA distributed software layer considers attribute values merely as a row of bytes which are not semantically interpreted by the RTI.

The flexibility of HLA gives the developers of distributed simulations freedom in defining their own standards for data exchange. The HLA Object Models introduce the possibility to reason on a more abstract level about simulation components, and thus make re-use possible. The Object Model Templates prescribe a formal language so that tools can be developed to handle FOMs and SOMs automatically. The use of HLA is greatly encouraged by making software such as an RTI and supporting tools freely available via the web [9].

Federation Development Process

While much of the early HLA development effort was focused on the creation of the run-time architecture, the importance of the early design processes is recognised. The US DoD has formalised a description for the high-level process by which HLA federations can be developed and executed to meet the sponsor's requirements. This model is known as the HLA *Federation Development and Execution Process*, or *FEDEP Model* [10].

The FEDEP Model has been influenced by the hands-on experience of HLA application developers. Essentially, it resembles a software engineering approach while also taking into account the necessary (scenario) preparation and execution steps for the actual simulation. The FEDEP model contains many subprocesses and relations between the processes and intermediate (software) products. However, at a more abstract level, it is possible to identify a sequence of basic steps that all HLA federations will have to follow for federation development and execution. These basic steps, which are collectively known as the Five Step Process, are listed below.

1. **Define Federation Objectives**. Define and document a set of needs that are to be addressed through the development and execution of an HLA federation, and transform these needs into a more detailed list of specific federation objectives.

2. **Develop Federation Conceptual Model**. Develop an implementation-independent representation of the real-world domain that applies to the federation problem space, and develop the federation scenario. In this phase also the federation objectives are transformed into a set of federation-specific requirements to use as success criteria during federation testing.

   From the perspective of Object-Oriented (OO) software designers, the federation conceptual model can be compared to a traditional object model. That is, the focus of federation conceptual model development is to identify federation objects, to identify static and dynamic relationships between object classes, and to identify the behavioural and algorithmic aspects of each class/object.

3. **Design and Develop Federation**. The purpose of the Design Federation sub-phase is to establish the membership of the federation, and to construct an engineering approach for developing and implementing the federation. The federation requirements, the federation scenario, and the federation conceptual model provide the necessary input. Federation Design also includes the development of a co-ordinated plan to guide the development, test, and execution of the federation.

   The purpose of the Develop Federation sub-phase is to develop the FOM, modify federates if necessary, and prepare the federation for integration and test (including database development, security procedure implementation, etc.). Notice that the FEDEP model does not prescribe the development of individual federates which could be taken from a simulation repository or completely be developed from scratch.

4. **Integrate and Test Federation**. Plan the federation execution, establish all required interconnectivity between federates, and test the federation prior to execution. The purpose of the test activity is to verify whether all participants can interoperate to the degree needed to achieve the federation objectives. Three levels of testing have been defined:

   Compliance Testing. Each federate is tested individually to ensure that the federate software correctly implements the HLA requirements as documented in the HLA Compliance Checklist [11].

   Integration Testing. The federation is integrated and tested as a whole to verify the data exchange between federates as described by the FOM.

   Federation Testing. The ability of the federation to achieve the federation objectives is tested. This includes observing the ability of federates to interact according to the defined scenario and to the level of fidelity required for the application.

   The first two testing activities are more concerned with testing of the basic communication ability of the federation ('syntax oriented') whereas the latter is focused on testing the semantics of the federation. In practise, however, there is also a need for testing the semantics of an individual *federate* with regard to its specified capabilities as described by the SOM. Once this is established, the number of (re)tests can be reduced for each (new) federation in which the federate will participate.

5. **Execute Federation and Prepare Results**. Execute the federation, process the output data from the federation execution, report results, and archive re-usable federation products.

The FEDEP model resembles a general software engineering approach, but it also explicitly acknowledges the existence of simulation component repositories to encorage the re-use of both models and implementations. Another important aspect is the integration with scenario development and associated data, and the subsequent execution and analysis steps. Nevertheless, the FEDEP model will have to be tailored to become a practical and beneficial ”tool” for both existing and new simulation developments.

Verification and Validation

Verification and *validation* are two important processes in software engineering. The difference between the verification and validation process can be meta-phrased as follows: the verification process is concerned with the question 'is the product being developed right ?' whereas the validation process is concerned with the question 'is the right product being developed ?'.

In traditional software engineering, a software product is validated against its well-defined (functional) requirements which should reflect the users' needs. In the simulation field, however, the issue of validation is much more complex, especially if some real-world phenomena is to be simulated. It is often unclear what aspects of the real world have to be modeled to obtain a simulation that is useful *for its intended purpose.* Moreover, the level of simulation *fidelity* is of great importance, i.e., the degree to which the simulation is an accurate representation of the real world. Fidelity concepts need to be understood well to cope with both the validation and the development of (distributed) simulations, as these are often a mixture of simulation components with diffent levels of fidelity [12]. Currently, there are no agreed upon definitions and metrics for the concept of fidelity, but there is an increasing scientific interest in the area of simulation fidelity.

The HLA community acknowledges the need for integrating the development process of federations with the Verification and Validation process. The FEDEP model defines a test- and integration phase; but Verification and Validation is a process that actually starts in the requirements-phase and is conducted throughout the life-cycle of the product, in parallel to the development process.

Each phase in the development cycle delivers one or more products (software, documents) which should at least be verified for completeness, consistency and correctness with respect to the products delivered in earlier phases of the development. Furthermore, conceptual models as well as the final products must be validated against the initial requirements and the intended purpose of the simulation with respect to the sponsor's needs. Procedures for verification and validation of both legacy simulations and newly developed simulations can be found in the VV&A recommended practices guide [13].

Verification and validation procedures are commonly used to identify and solve problems in a particular product, but they can also be used to merely assess, and document, the quality of a product. This documentation is actually the key to the success of the 're-use' of simulation components: it is often the primary source of information on which the decision is based whether the component can be re-used as part of some other product. A well-defined verification and validation policy, in conjunction with a well conducted configuration management policy, is the most prominent precondition for the success of simulation component repositories and the concept of re-usability at large.

Current Research

HLA is a technical framework that enables the development of large-scale distributed simulations in which many federates can participate in the same scenario. Large-scale distributed simulations potentially suffer from network related problems, such as latencies and limited bandwidth. The RTI offers two categories of management services (DDM and Ownership Management) to cope with these aspects. These services should be made accessible through a software middleware layer such that application developers do not need to know all related technical intracacies.

DDM and Ownership Management related issues are currently being investiged by TNO Physics and Electronics Labatory (TNO-FEL) in co-operation with Eindhoven University of Technology (TUE). The research involves the construction of a distributed simulation model to measure the performance capabilities of the available RTI and to evaluate design trade-offs involving the use of these two RTI management services.

Further research interests include the formalisation of Conceptual Model descriptions in a uniform manner for subsequent storage in a Conceptual Model repository. Such a repository is mandatory for effectuating the re-use of simulation components. These Conceptual Model descriptions, or templates, will also help the system analyst or developer to structure the required information about the conceptual model. Furthermore, formal descriptions are necessary to construct automated code generators that can help in transforming high level descriptions into code skeletons for a specific implementation language.

Applying the FEDEP model in conjunction with Verification and Validation is another research topic. The FEDEP model is quite generic and should be tailored to a specific application area. Especially the research to the processes of transforming so-called *legacy* simulations into HLA compliant simulations with well-defined levels of fidelity may offer very practical results.

Finally, the construction of middleware layers and standardised architectures for the development of individual federates are of great importance. HLA does not prescribe how an individual simulation or simulator should be built; it merely provides a technical framework with the necessary run-time management services. Within the Netherlands HPCN project SIMULTAAN the concept of re-usable distributed simulator components is investigated and applied to improve future co-operation on simulator development [14, 15].

Application Areas

From the origin of HLA it may be concluded that most applications are found in the military domain. We mentioned team training and mission preparation. To support its research to these application areas, TNO-FEL has developed the Electronic Battlespace Facility (EBF) [16]. The EBF offers an infrastructure to create synthetic environments for experimental research to real-time distributed military simulations. The EBF consists of hardware and software components such as graphical computers, simulator mock-ups and supporting tools, that can be networked with external simulators and mock-ups to form a project-specific federation. These application-specific federations can then be used for simulations in the field of operational analysis, mission preparation/rehearsal and procedural training.

Application areas for distributed simulations are not only found in the military domain, but also in the civil and space domains [17]:

Collaborative Virtual Environments. Currently the most advanced 3D user interfaces to collaborative distributed simulations are immersive virtual environments. Immersive VEs pose severe communication requirements due to the nature of the generated network traffic. Additionally, there are a number of critical human factors that need to be assessed before this technology can be successfully applied. The EU ACTS project COVEN [18], in which TNO-FEL participates, aims to develop a software platform for teleworking and virtual presence applications. Network trials are being conducted to investigate communication aspects under different usage conditions [19]. Furthermore, a study to the feasibility of using HLA concepts to support interoperability between COVEN applications is being done by TNO-FEL.

Industrial applications. To date product development requires precise planning and reduced cycle times. The trend towards global collaboration induces an increase in interoperability requirements between information systems of a world-wide operating *virtual company*. Collaborative decision making amongst geographically dispersed business operators requires a shared situation awareness, built from consistent, up-to-date, reliable and authoritative information. Distributed simulation facilities will play a key role in such collaborative decision aids. The HLA technology is the prime candidate as supporting technology for such applications [20].

Driving simulation. The automotive industry generally recognises the potential value of driving simulation for safety enhancement, training, and as a research tool. To date however, driving simulators have mainly been employed in the latter area. The potential benefits of networking several driving simulators into one synthetic environment has been recognised and is further explored in several international research projects [21].

Civil Aviation. Driven by its large installed base of simulators and the trend towards international collaboration, and even globalisation of the aviation industry, the civil aviation community has taken an active interest in simulator interoperability. The US Federal Aviation Administration (FAA) has, within the framework of its National Simulation Capability (NSC) program, developed a Formal Analysis Process (FAP), which is an analysis environment, based on HLA concepts, but adapted to meet a number of civil aviation requirements, to support R&D efforts aimed at improving traffic flow, enhancing safety, increasing capacity, and improving overall system performance [22].

International space missions. The European Space Agency (ESA) currently assesses HLA as a technology to accomplish a more effective use of available facilities, mainly simulators and simulation models, that are located throughout Europe. International co-operations such as the International Space Station (ISS) require standards for distributed simulations and model exchange. To demonstrate the possibilities of DIS and HLA, ESA developed in co-operation with the russian Gagarin Cosmonout Training Centre (GCTC) a prototype application for the rendez-vous and docking of the Automated Transfer Vehicle (ATV) and the ISS [23].

Conclusion

Although HLA is primarily aimed at military applications, and mainly driven by the US Department of Defence, it is both general applicable and available outside the military domain. Applying HLA by non-military institutions is greatly encouraged by DMSO. We have mentioned a few application areas in the civil and space domains, but more can be thought of.

The question is whether HLA is as useful as promised by their developers. To answer that TNO-FEL has developed some prototype applications to build up experience. TNO not only applies HLA as a given technology but also provides feed-back to the HLA community, and DMSO in particular, based on experimental research. Examples are the proposed middleware layer, the research to fidelity issues, and the application of the FEDEP model in relation to VV&A. The current RTI as made available by DMSO is investigated in co-operation with Eindhoven University of Technology. Suggestions for improvements on the DDM services have been communicated to DMSO and will be taken into account when developing the next version of the RTI.

HLA is part of a larger standardisation effort in the US on Modeling & Simulation. As such, HLA is promising: not only the run-time communication is addressed, but also the software engineering process is taken into account. For military applications the usage of HLA is greatly encouraged by the US DoD. In fact, a large simulation community is more or less dependent on the technical progress of HLA and in particular on the development of the RTI. Unfortunately, a stable version of RTI which implements all services efficiently is not yet available. Besides, using the RTI requires quite some programming effort due to the administration of objects and attributes. To cope with the latter, users are developing middleware layers and code generators to tailor HLA to their own application areas. As a standardisation effort in Modeling & Simulation, HLA is promising and should be applied with care.

References

[1] DIS Steering Committee, *The DIS Vision, A Map to the Future of Distributed Simulation*, Version 1, Institute for Simulation & Training, May 1994, Orlando, Florida, USA, IST-SP-94-01.

[2] R. Weatherly, D. Seidel, and J. Weismann, *Aggregate Level Simulation Protocol*, Proc. 1991 Summer Comp. Sim. Conf., pp. 953-958, Baltimore, July 1991.

[3] Simulation Interoperability Standards Organization, http://www.sisostds.org

[4] IEEE P 1516.1, *Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) - Federate Interface Specification*, April 1998.

[5] IEEE P 1516, *Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) - Framework and Rules*, September 1998.

[6] IEEE P 1516.2, *Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) - HLA Object Model Template (OMT)*, February 1998.

[7] US Department of Defence, *High Level Architecture Run-Time Infrastructure Programmer's Guide 1.3 Version*, 31 july 1998.

[8] Katherine L. Morse, *HLA Data Distribution Management: Design Document*, Version 0.7, 12 November 1997.

[9] HLA Homepage, http://hla.dmso.nl

[10] Robert R. Lutz, *FEDEP V1.1*, Proc. 1998 Spring Simulation Interoperability Workshop, Orlando, FL, March 1998.

[11] Defence Modeling and Simulation Office, *HLA Compliance Checklist*, Version 1.3 (Draft), May 1998.

[12] Manfred Roza, Paul van Gool, Hans Jense, *A Fidelity Management Process Overlay onto the FEDEP Model*, Proc. 1998 Fall Simulation Interoperability Workshop, Orlando, FL, September 1998.

[13] Department of Defence, *DoD Modeling and Simulation (M&S) Verification, Validation, and Accreditation (VV&A)*, April 29, 1996.

[14] Nico Kuijpers, Paul van Gool, Hans Jense, *A Component Architecture for Simulator Development*, Proc. 1998 Spring Simulation Interoperability Workshop, Orlando, FL, March 1998.

[15] SIMULTAAN Homepage, http://www.nlr.nl/public/hosted-sites/simultaan

[16] Hans Jense, Nico Kuijpers, Robert-Jan Elias, *Electronic Battlespace Facility for Research, Development and Engineering*, Proc. 1997 Fall Simulation Interoperability Workshop, Orlando, FL, September 1997.

[17] G.J. Jense, N.H.L. Kuijpers, A.C.M. Dumay, *DIS and HLA: Connecting People, Simulations and Simulators in the Military, Space and Civil Domains*, 48th International Astronautical Congress, Turin, Italy, October 6-10, 1997.

[18] COVEN Homepage, http://coven.lancs.ac.uk

[19] Chris Greenhalgh, Steve Benford, Adrian Bullock, Nico Kuijpers and Kurt Donkers, *Predicting Network Traffic for Collaborative Virtual Environments*, Computer Networks and ISDN Systems, 30 (1998), 1677-1685. First appeared in Proc. TNC'98, TERENA Networking Conference, Dresden, 5-8 October 1998.

[20] David Payne, *Interoperable Systems for 21st Century Logistics: Training, Planning, and Execution*, Proc. 1997 Spring Simulaton Interoperability Workshop, Orlando, FL, March 1997.

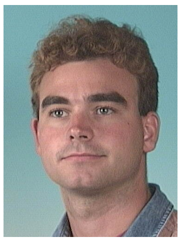
[21] Reginald T. Welles and Darrell R. Turpin, *When worlds collide - Considerations for integrating a driving simulator into a distributed interactive simulation network (DIS)*, Proc. 14th DIS Workshop, Orlando, FL, pp. 865-874, 1996.

[22] Paula Nouragas et.al., *A formal analysis process based on the High Level Architecture*, Proc. 1997 Spring Simulation Interoperability Workshop, Orlando, FL, March 1997.

[23] Vankov et.al., *Distributed Interactive Simulation of Rendezvous and Docking with International Space Station*, Proc. 1997 Fall Simulation Interoperability Workshop, Orlando, FL, September 1997.

About the authors

Marco Brassé is a member of the scientific staff in the Command & Control and Simulation Division at TNO-FEL. He is a software architect for several projects in the area of distributed simulation, both nationally and internationally with partners in the military and simulation industry. He holds a M.Sc. in Computing Science and a Master of Technological Design in Software Technology, both from Eindhoven University of Technology.



Nico Kuijpers is assistant professor at the Department of Mathematics and Computing Science of Eindhoven University of Technology (TUE). His research interests are medical image analysis, large-scale (biomedical) simulations and distributed simulations. In his former job at TNO-FEL he was a member of the scientific staff in the Command & Control and Simulation Division. He was the project leader for several collaborative projects in the areas of Collaborative Virtual Environments and distributed simulation, both nationally and internationally with partners in the industry and acadamic world. He holds a M.Sc. in Computing Science and a Master of Technological Design in Software Technology, both from Eindhoven University of Technology.