Proceedings of the 2015 Winter Simulation Conference
L. Yilmaz, W. K. V. Chan, I. Moon, T. M. K. Roeder, C. Macal, and M. D. Rossetti, eds.

TOWARDS AUTOMATING THE DEVELOPMENT OF FEDERATED DISTRIBUTED SIMULATIONS FOR MODELING SUSTAINABLE URBAN INFRASTRUCTURES

Ajitesh Jain
Richard Fujimoto
Jongchan Kim

School of Computational Science & Engineering
Georgia Institute of Technology
Atlanta, GA 30332, USA

Mengmeng Liu
John Crittenden
Zhongming Lu

School of Civil & Environmental Engineering
Georgia Institute of Technology
Atlanta, GA 30332, USA

ABSTRACT

The study of sustainable urban systems requires analysis of interdependencies and relationships among infrastructures and social processes. Federated simulations using the High Level Architecture is a natural approach to modeling such systems-of-systems. Simulations interoperability requires a common structure and meaning of shared data represented in a Federation Object Model (FOM). Developing the FOM and modifying simulations to comply with the FOM requires a significant amount of time and effort. We describe a system to automate portions of this task. Specifically we present a workflow by which existing simulations can be integrated in a semi-automated process to reduce the manual labor required. We use SysML to describe entities of the federated distributed simulation. These descriptions are used for automatic generation of the FOM and code required for HLA integration. Finally we present a case study applying this methodology to create a federated distributed simulation to study sustainable urban growth.

1 INTRODUCTION

The development of more resilient and sustainable cities is an area of increasing concern. Cities are undergoing a revolution with new developments such as the increased deployment of electric and autonomous vehicles, widespread instrumentation providing detailed information concerning energy usage, growing use of renewable energy sources relative to consumption of fossil-fuels, and new modes of interaction and information propagation via social networks, to name a few. It is well understood that it is not sufficient to study individual infrastructures (e.g., transportation, water, energy, telecommunications, food production) or social elements (e.g., economics, social attitudes concerning the adoption of more sustainable practices, urban growth, land use) of modern cities in isolation because there are important interdependencies that must be considered. For example, one recent study notes that increased use of electric vehicles that rely on coal generated electricity can *increase* the amount of pollution produced compared to conventional vehicles (Tessuma, Hillb, and Marshall 2014). Energy production relies heavily on water for cooling in fossil fuel and nuclear power plants as well as for hydro-electric power generation. In many cases these uses compete for the same water as agriculture and food production industries, with impacts that affect the cost-of-living for consumers and their ability to purchase technologies that can lead to more sustainable growth and development. While the complex systems arising within each sector such as transportation, water, energy, and food productions are understood to varying degrees, *systems of complex systems* are very poorly understood, and it is clear that modeling and simulation will play a critical role in studying their behavior. The importance of interactions among infrastructures has similarly been recognized in creating cities that are more resilient

to natural and human-caused disasters and emergencies. Integration of infrastructure simulations remains an important challenge today (Adam et al. 2013).

The most well-known approach to creating federated simulations is the High Level Architecture (HLA), IEEE Standard 1516 (IEEE Std 1516-2010 2010, IEEE Std 1516.1-2010 2010, IEEE Std 1516.2-2010 2010). One of the aims of HLA is to support interoperability and reuse among simulations. The Federation Execution and Development Process (FEDEP) is a recommended IEEE procedure for creating distributed simulations using the HLA (IEEE Std 1516.3-2003 2003). The federation design process involves identifying existing, reusable federates that meet or partially satisfy the federation requirements and modifying them accordingly. The next step involves creating a Federation Object Model (FOM) that describes all shared information (objects, attributes, associations, and interactions) essential to a particular federation. Also each federate should be modified so that it complies with the shared object model. Creating this shared data contract and modifying the existing simulations to conform to this contract requires a significant amount of time and effort for the developer. In this paper, we describe a system and methodology by which the amount of time and manual effort required for this process can be reduced, thereby increasing developer productivity.

In order to automate the generation of the FOM, the developer must identify the information each simulation is able to import and export and document this information in a Simulation Object Model (SOM). In the next step, the developer must specify a common federation-wide data model in the FOM. One of the issue that often arises when reusing existing simulations is that disparate representations of a common entity may have been created. For instance, one of the simulations might specify a class called *car* in its object model while another uses a class called *vehicle*. In order to address this issue a shared vocabulary and conceptualization of simulation entities is necessary to enable unambiguous communication of concepts. Ontologies provide a formal means to define common conceptualization as well as terminology and representation of knowledge, and potentially could be useful for semantic analysis and inference. We make use of concepts utilized by ontologies in our work. Another issue that often arises while developing the FOM is the use of different units of measurements or data types for the information exchanged between federates. To address this, the developer must explicitly create code to implement the transformation from one type to another.

Our approach involves first specifying a model in a common modeling language, and then using software to automate the transformation of the model into federated simulation code. We use SysML which is based on the Unified Modeling Language (UML) to describe the model (Delligatti 2014). SysML is a well-known graphical language used for model-based system engineering and widely used in industry for the last several years. It supports a wide range of lifecycle processes such as specification, analysis, design, verification and validation of systems in the model-based systems engineering approach. Using SysML for ontology creation circumvents the need to learn other ontology development tools (Rouquette and Jenkins 2010). This approach also helps to increase the extensibility and flexibility of the federation as the individual federates can be modified without worrying about regenerating the FOM and modifying other federates as the process for the FOM and code generation is automated.

2 RELATED WORK

Issues in interoperability for simulations have been studied by various research efforts in the past. Hofmann (2004) describes some of the technical, semantic and pragmatic challenges involved in coupling model-based information systems. He emphasizes that the most difficult part of developing simulation models lies in the conceptual modeling phase where one must develop “appropriate models.” Appropriate models are informally defined as abstractions and documentation of the assumptions used by the model. One of the problems is that of hidden assumptions that arise when the assumptions used by different models are not properly documented and the federated simulation produces incorrect results during the execution. Wang, Tolk, and Wang (2009) further describe the importance of conceptual modeling. They propose a Levels of Conceptual Interoperability (LCIM) framework that can be used to develop

conceptual models to achieve different levels of interoperability. The authors describe seven levels of interoperability and describe the requirements and information to be defined to reach each level. These seven levels of interoperability are depicted in Figure 1. They advocate the usage of the LCIM framework to improve the levels of interoperation supported by the High Level Architecture (HLA) in general and by the Base Object Model (BOM) in particular. BOM is a piece-part of the conceptual model, SOM, or FOM that can be used as a building-block to construct simulation and federation object models.

For simulations to be interoperable they should be conceptually modeled so as to achieve at least level 3 of the LCIM. In order to achieve this level the meaning of the data and the structure of the data should be clearly specified. One of the approaches to capture the meaning of data is by the use of ontologies. Rathnam and Paredis (2004) describe the use of an ontology-based framework for modeling federates. They make use of relationships between the SOM of federates captured using ontologies to generate the Federation Object Model (FOM). Also Rathnam (2004) presents an algorithm termed the Graph based Inference of Transformations (GRIT) for automatic generation of a common representation and transformation routines for different data types. We leverage the GRIT algorithm to automate the generation of the FOM. Work by Benjamin and Akella (2009) further describes how ontologies can be used by the modeling and simulation community to solve the problem of simulation interoperability.

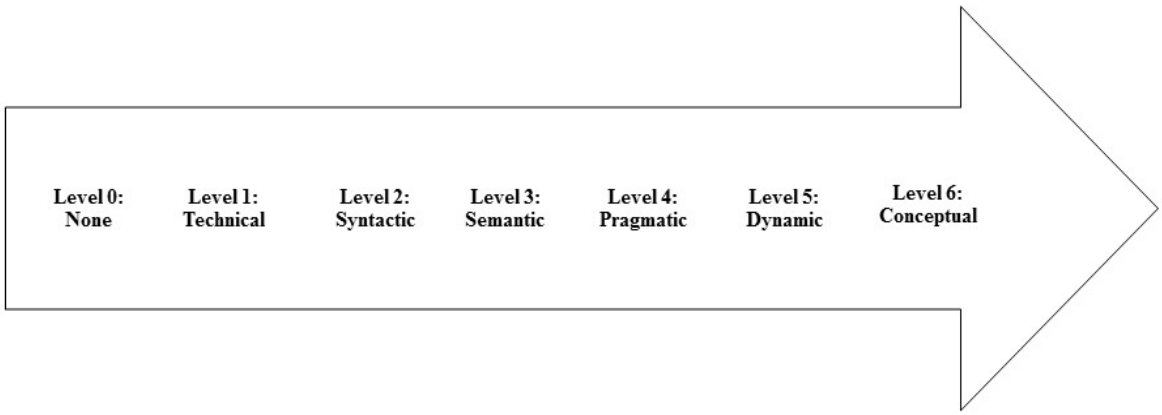

Figure 1: Levels of conceptual interoperability.

McGinnis et al. (2011) advocate the use of OMG SysML as a modeling language for simulations because it not only supports defining an ontology, but it also supports developing specific models using the ontology. This complete integration between defining and using the ontology helps in faster development and leads to implementable Domain Specific Languages (DSL). Huang, Ramamurthy and McGinnis (2007), and Schönherr and Rose (2009) describe an approach for generating code from SysML descriptions of the model. They present a workflow where once the model has been described in SysML, one can use the XMI export feature to obtain the equivalent XML representation which can be parsed using standard XML software such as XPATH or JAVA SAX parser to generate code. XMI is an OMG standard for representing SysML models.

Bocciarelli et al. (2013) describe an automated framework to build and execute distributed simulations. The authors describe the *HLACloud* framework that enables automated generation of JAVA/HLA code and deployment scripts from SysML models. They make use of a *SysML4HLA* profile that extends SysML's vocabulary to include HLA related concepts such as federate, interaction, object attributes and parameters, etc. We use some of these concepts in our work for developing SysML models and automating the generation of code.

There is a growing body of work focusing on modeling and simulation tools for studying interacting urban infrastructures. A survey of work in this area is presented in Ouyang (2014). Most relevant to the

work described here is the approach described in Casalicchio and Galli (2008), and Casalicchio et al. (2011) where XML descriptions are proposed for creating federated simulations.

3 SYSTEM DESCRIPTION

The workflow for developing automated FOM and code from existing simulation models is depicted in Figure 2. The workflow consists of the following principal steps each of which is described next:

1. Creating a SysML description of the individual simulators (SOM) that are to be integrated
2. Define relationships among similar entities across the federation
3. Create the FOM and associated translation routines
4. Generate code for the federated simulations from the FOM
5. Integrate code to construct the federated simulation

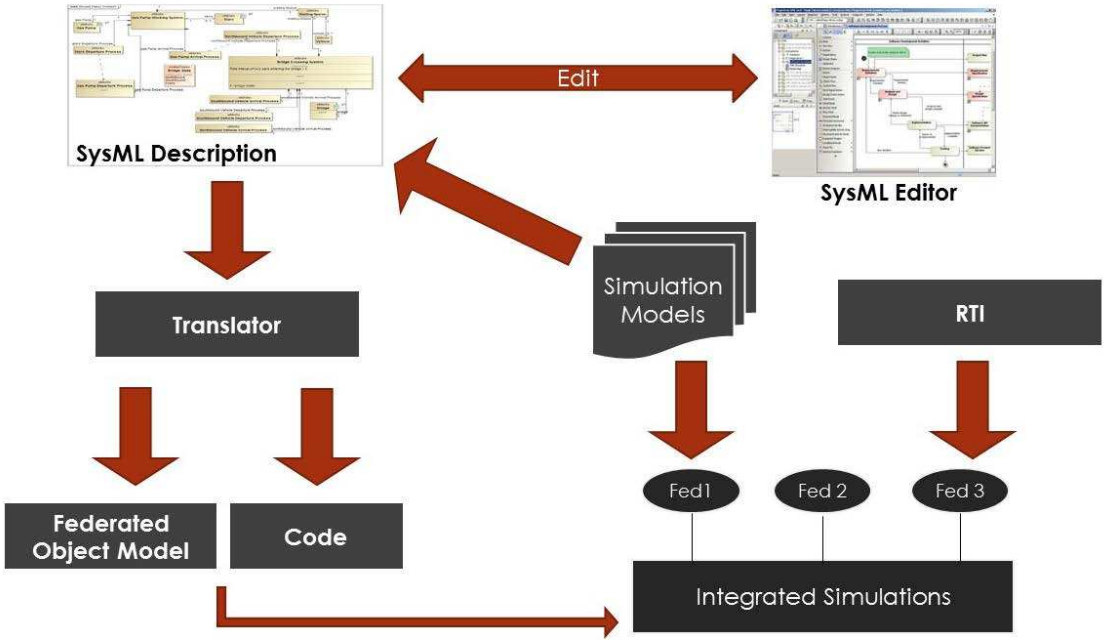

Figure 2: System description.

3.1 Create SysML Descriptions of Simulations

The SOM for a federate describes objects, attributes, interactions and parameters in a particular simulation that can be used externally in a federation. The developer should specify the SOM using SysML constructs such as the block definition diagram and parametric diagram. While developing SOMs represents a significant amount of effort, once completed, these specifications can be reused as the simulation is used in other federations.

There are many advantages of using SysML to represent the system of interest. First, diagrams written in SysML provide a good, standardized communication tool among stakeholders from different disciplines. Second, as the syntax and semantics of SysML are (semi) formally defined, diagrams written in SysML are computationally interpretable. Third, we can represent various aspects of models in SysML independent of the programming language in which they are implemented. Fourth, SysML provides a new

diagram type called parametric diagram that is not present in UML. Parametric diagrams can be used to create systems of equations and constrain the properties of blocks. Finally, design tools using SysML are readily available that greatly simplify maintenance of the description as various aspects of the simulator change over time.

We use stereotypes to extend the vocabulary of SysML to HLA concepts such as attribute, federation, federate, publish/subscribe, etc. The stereotypes depicted in Figure 3 are minimal and include only the properties we required at the time of this writing. This set of stereotypes is a subset of the *OMTKernel* package introduced by Bocciarelli et al. (2013).

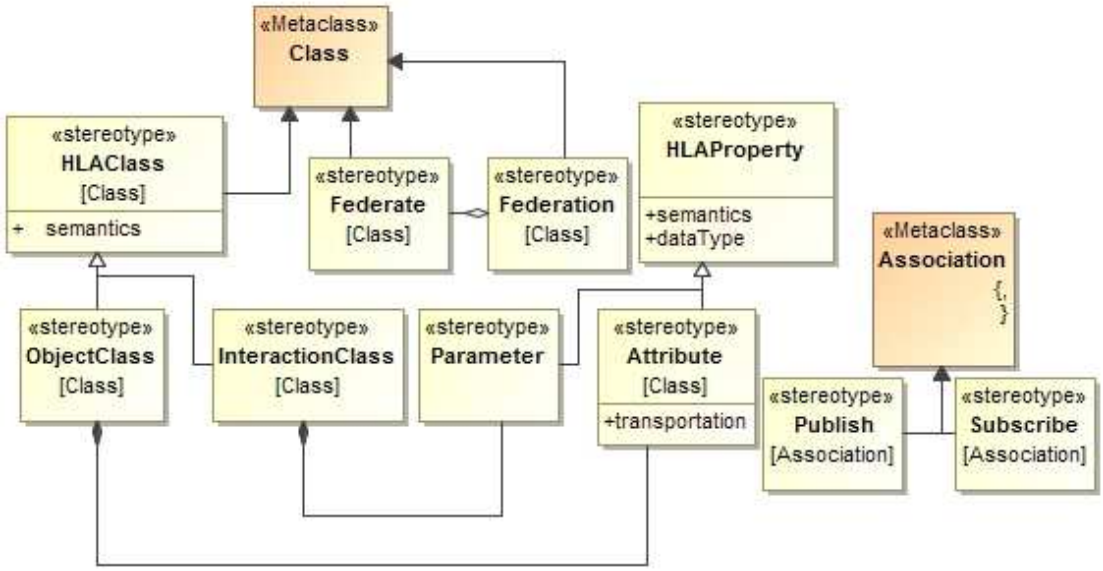

Figure 3: HLA stereotypes.

3.2 Define Relationships Among Similar Entities Across the Federation

The relationship among attributes can be specified by an association block as shown in Figure 4. This method of representation builds upon the work described by Tarun and Paredis (2004). That work used Protégé for ontology development rather than SysML. The Transformation block in Figure 4 contains the routine to convert data from one entity attribute to another. It also indicates whether the transformation is lossy, i.e., there is some loss of information during the conversion process. For instance if one of the attributes indicates the 3D location of a vehicle while another attribute only indicates a 2D location then transforming from 3D to 2D results in a loss of information and such a transformation would be marked as lossy. Figure 5 shows an example representation of radius and diameter entities of a model.

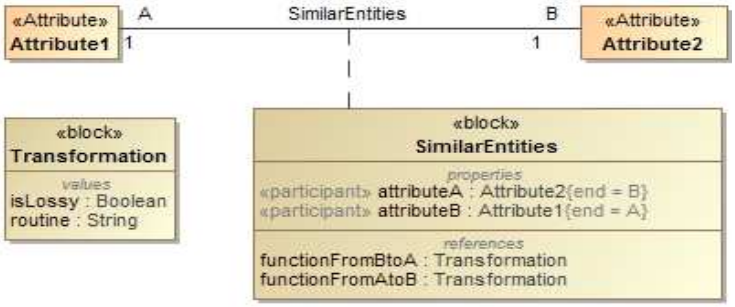

Figure 4: Representing relationship between similar attributes.

If there are a large number of simulations or if the simulations contain many attributes or parameters, it will become difficult for the developer to check for similar entities across the simulations. In order to aid in this process, we have developed a similarity detector script in Python. This script takes as input the attribute names in all simulations and produces as output the similarity scores between attributes belonging to different simulations. The developer may view the similarity scores for attributes for each pair of federates. The similarity detector script makes use of the dictionary meaning of the attribute name and uses Natural Language Toolkit (NLTK) to find similarity measures. NLTK is a toolkit used to develop natural language processing programs in Python. We use Wordnet, a lexical database for English that organizes words into hierarchies. The Wordnet APIs provide many similarity measures to detect the closeness of two words (Rodriguez and Egenhofer 2003).

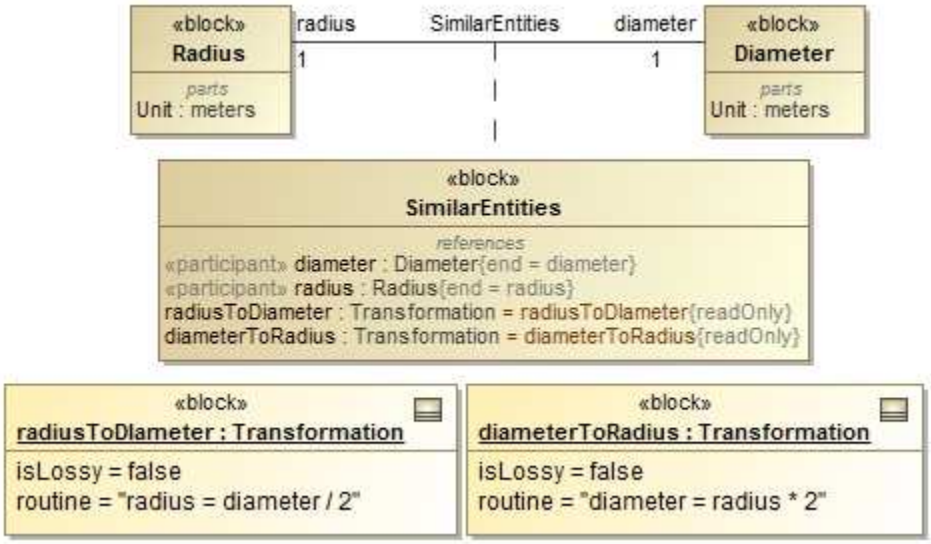

Figure 5: Representing relationship between radius and diameter.

3.3 Generate FOM and Transformation Routines

Once the developer has defined relationships, the SysML description is exported as XMI and is input to the translator program. The translator program is written in JAVA which uses the XML DOM parser to generate an internal model. This internal model represents a weighted directed graph where the nodes represent the attributes in the model and the edges indicate similarity between the attributes. Each edge is assigned a weight of (2 * number of edges) or 1 indicating if the transformation from source vertex to destination vertex is lossy or not, respectively. In order to generate the FOM, we leverage the GRIT algorithm developed by Rathnam (2004). The GRIT algorithm is a graph based algorithm to find a common representation among similar entities and generates transformation routines for conversion from this common representation to simulation entities and vice-versa. It takes as input the aforementioned graph and finds the shortest distance and shortest path between every pair of connected vertices using Dijkstra's shortest path algorithm. The algorithm chooses that entity as a common representation such that the number of lossy conversions from a common entity to a simulation entity and vice-versa are minimized. Once the common representation has been determined, based on the shortest path from an entity to its common representation, it generates the transformation routines for converting between the two. While generating the transformation routines, it checks for the measuring units of the two attributes and takes into account any explicit conversion routines specified in the Transformation block of Figure 4.

3.4 Generate RTI Code from FOM

After the FOM has been created in the previous step, we generate RTI code for each of the federates. We use the Federated Simulations Development Kit (FDK) developed by Fujimoto et al. at Georgia Tech that implements a subset of the HLA Interface Specification. The output of this step includes generation of several components. The first is the FED file that specifies the class hierarchy used by the federation. This file is used by the RTI to set up the class-based publications and subscriptions using the HLA's declaration management services. Further, routines for sending and receiving data values using the Update Attribute Values and Reflect Attribute Values services are created. Code for invoking time management services are created to synchronize the federation. Finally initialization routines are created for each federate for tasks such as registering the federate and instantiating the federation, as well as invoking declaration management services as noted earlier. Once generated, these code segments must be manually inserted into each of the federates by the developer, as described next.

3.5 Integrate Existing Model Code With RTI , FOM and Generated Code

Once the code and the FOM have been generated, each federate must be modified manually to integrate the various routines generated in the previous step with the original simulators to create the federated simulation. This includes determining where attributes must be sent (update) and received (reflected), insertion of the routines to invoke the time management services as well as incorporation of initialization routines.

A number of technical issues must also be addressed to integrate the federates with the RTI, especially if the federates and RTI are developed using different programming languages. There is a substantial body of work and a number of technical approaches that have been developed to address this problem. For example, CORBA is a standard defined by the OMG to facilitate interaction between software running on different platforms and languages. CORBA uses an interface definition language (IDL) to specify the interfaces that object presents externally. CORBA then specifies a mapping from IDL to a specific implementation language such as C++ or Java. Another approach is to use web services. Web services allow electronic devices to communicate over a network. SOAP is a protocol that utilizes XML for information exchange for implementing web services. Wu et al. (2007) describe how to couple simulations with an RTI using web services. A third approach to couple Java and C/C++ code is the Java Native Interface (JNI). JNI allows one to call native modules from JAVA code. This can be used if RTI software is in C/C++ and simulation software is in JAVA or vice-versa. Finally, interprocess communication could be accomplished by transmitted data through files. A daemon can be written in the same language in which the RTI is written that continuously polls specific files for input and outputs to specific files. Similarly, the model needs to be modified to input and output the required attributes to specific files. Similarly, sockets could be used for communications rather than files. These mechanisms represent various tradeoffs among ease of implementation and execution efficiency. Our initial prototype uses files, but other options are being considered as future work.

4 EXAMPLE: SUSTAINABLE URBAN SYSTEMS

The growth of cities is driven by interactions among infrastructures and the social-economic environment. Infrastructures are the foundation for residence, business and mobility in cities. Access to roads, water, sewer, and energy influence the creation of new social-economic developments. Economic growth and social harmony in turn create new demands for the improvement of infrastructure services creating a feedback loop. In order to study the interactions among different infrastructures we developed a distributed federated simulation using the integration methodology described in the previous section. One of the federates is an agent-based model described in Lu (2014) that models social preference and adoption of green infrastructure designs such as low-impact development strategies to control storm water. This model also considers how emergent properties such as mean home prices, water usage, tax

revenue etc. vary over time. A transportation simulation was integrated with this model that is used to determine bus routes and the location of bus stops in a developing area. The integration of the two models helps to determine how bus routes might influence housing prices and how that will affect the decisions of homebuyers and developers and ultimately, the spatial pattern of urban growth.

4.1 Urban Growth Model

The urban growth model uses an agent-based approach to study how the interactions of home buyer, developer, and the local government at the neighborhood level influence the adoption of single family versus apartment homes and how they impact government policy. The interactions among these agents lead to changes in city properties, i.e., land use patterns. Homebuyers choose to purchase single-family houses or apartment homes. The government provides infrastructure services and improvements. The developer converts the undeveloped land into residential subdivisions to satisfy housing demand and sells the properties to homebuyers. The various agents and their interactions are shown in Figure 6.

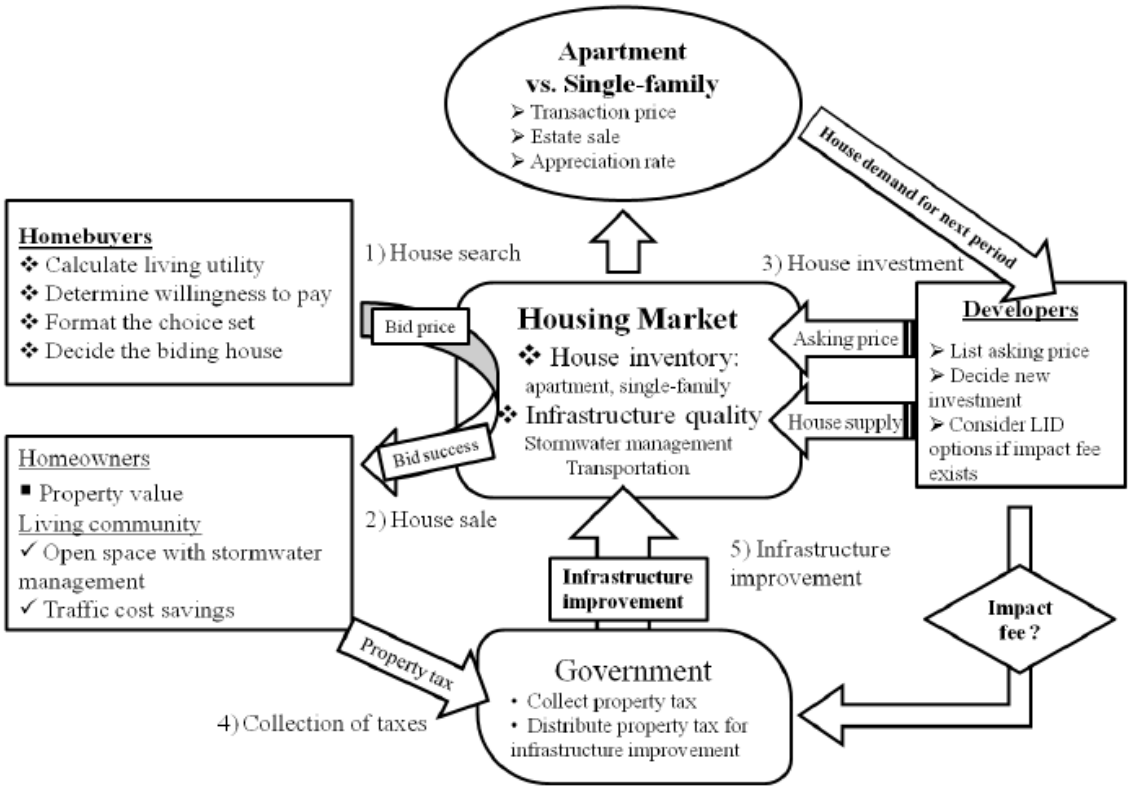

Figure 6: Schematic diagram of the agent-based urban growth model (Lu 2014).

4.2 Transportation Model

Transit-oriented development (TOD) is an example of a strategy for green infrastructure design which aims to maximize access to public transport. TOD helps to reduce dependence on private vehicles and hence can be effective in reducing traffic congestion and fuel emissions. Bus transportation is one of the most prevalent forms of public transportation. Therefore, transit-oriented development of bus services in a city is of vital importance. A number of factors such as the location of bus stops, residential population near bus stops, bus routes, ticket fares etc. need to be considered in planning the bus transportation system in a city. This transportation model is based on work described by Feng, Hsieh and Peng. (2010) for

finding the optimal bus routes that maximizes the profits realized by a bus service company. The model simulates how the bus routes change with the increasing population of a region. It assumes a rectangular plot of land that is divided into rectangular patches. Roads run between plots and a bus stop can be constructed at any of the intersections. It also simulates the traffic costs of residents living in that region.

4.3 Integration

The transportation model receives as input the population distribution from the urban growth model. It computes bus route and simulates the transit system in order to determine traffic cost savings by residents in that area over time. This traffic cost savings from the transportation model is fed back to the urban growth model which then simulates urban growth, determining the spatial pattern of urban growth over time. This interaction is shown in Figure 7.

The methodology described earlier was used to integrate these two simulations. We began by developing the descriptions of data exported/imported by the two models in SysML, i.e., the SOM for each simulator. The SysML descriptions of the urban growth model and the transportation model are shown in Figure 8. We have omitted details from the diagrams for the sake of clarity. In the next step, we defined the relationships between the two SOMs. We observed that the two models had traffic cost and land patch as common entities and hence connected the two using the "similar Entities" relation. This SysML file was exported as XMI and used as input to the translator program. The translator program generated the code and the FED file required for RTI integration.

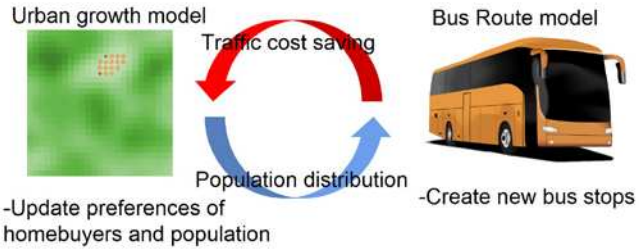

Figure 7: Flow of information between the urban growth model and the transportation model.

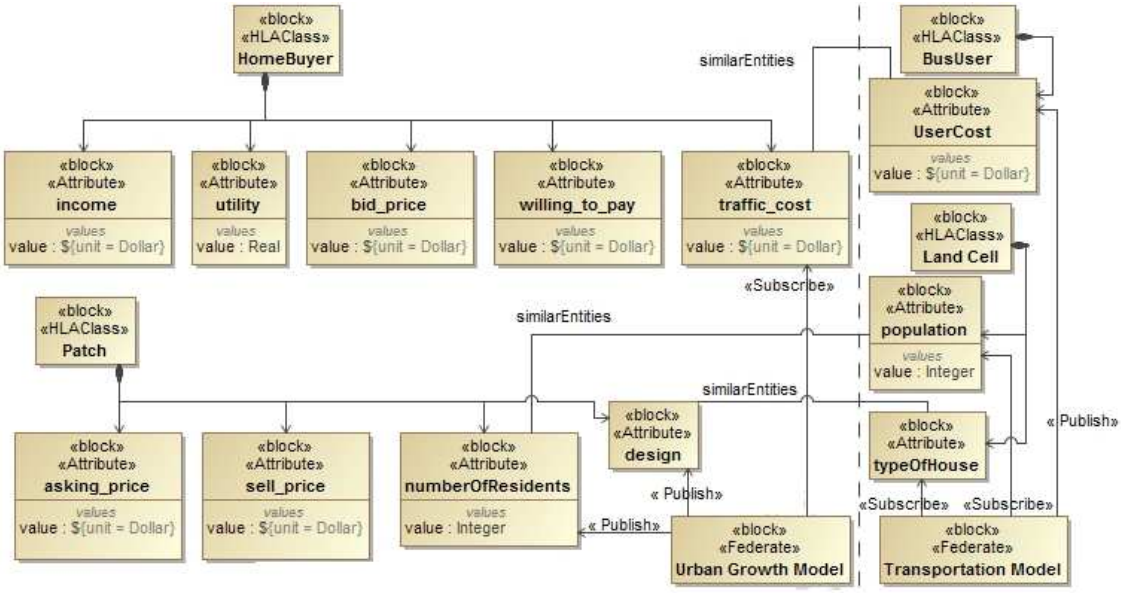

Figure 8: SysML descriptions of the urban growth model and the transportation model.

In the last step we manually modified the two simulation programs to insert the routines generated by the translator program. Integrating the transportation model with the RTI was straightforward as both are written in C++. However, the urban growth model was written using Netlogo. To integrate this model with the RTI we developed a simple interprocess communication service using files to translate information. Read file and write file methods were added to the simulation program that were invoked at appropriate points in simulation time. Also a daemon process was developed in C++ that could read and write to the same files. This enabled integration of the urban growth model with the RTI, although with a certain performance penalty. Other approaches to improve this integration are under investigation.

4.4 Results

Simulation results from the transportation model indicate that the availability of bus service in an area increases the traffic cost savings as shown in the left graph of Figure 9. When the traffic cost savings increase the willingness of homebuyers to pay also increases and this results in the increase of mean house prices. The effect of availability of bus service in an area on the house price obtained from the urban growth model is shown in the right graph of Figure 9.

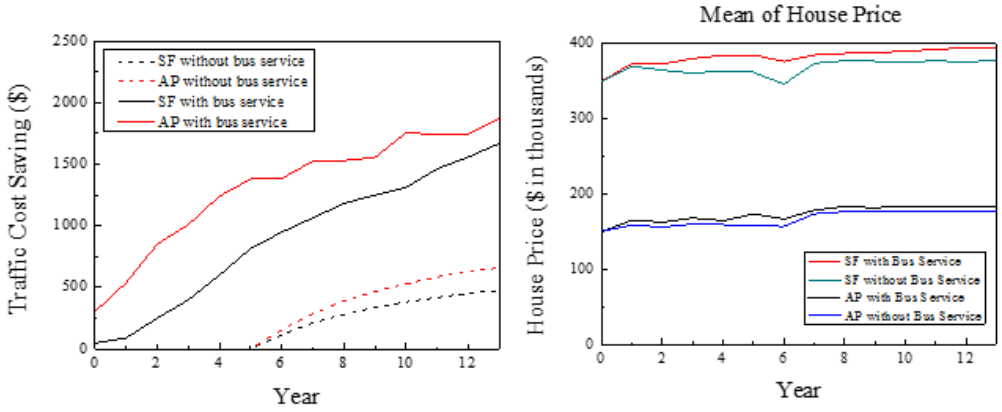

Figure 9: Variation of traffic cost savings over time from the transportation model (left) and variation of mean house prices over time from the urban growth model (right) for single family (SF) and apartment (AP) houses.

5 CONCLUSION

HLA provides a capability to reuse existing simulations. However modifying existing simulations and generating a common FOM can be a time consuming and error-prone task. In this paper we presented a methodology by which one can automate the creation of the FOM and HLA code and hence accelerate the development process for creating federations. Also we presented an example using this methodology for an application concerning sustainable urban growth. The approach described here is not restricted to modeling urban infrastructures nor limited to studying sustainability issues, however, work in these domains are being used to evaluate the system.

A proof-of-concept prototype of the federated distributed simulation system using the automated tools has been created. The example described here is being used to demonstrate the viability of the workflow structure. Future work on this project will focus on incorporating more complex simulation models into the system and utilizing the tools to better understand issues and develop approaches to steer urban infrastructures toward more sustainable practices for growth.

ACKNOWLEDGMENTS

Support for this research was provided by the National Science Foundation under Grant 1441208.

REFERENCES

Adam, N., R. Stiles, A. Zimdars, R. Timmons, J. Leung, G. Stachnick, J. Merrick, R. Coop, V. Slavin, T. Kruglikov, J. Galmiche, and S. Mehrot. 2013. "Consequence Analysis of Complex Events on Critical U.S. Infrastructure." *Communications of the ACM* 56:83-91.

Benjamin, P. and K. Akella. 2009. "Towards Ontology-Driven Interoperability For Simulation-Based Applications." In *Proceedings of the 2009 Winter Simulation Conference*, edited by M. D. Rossetti, R. R. Hill, B. Johansson, A. Dunkin and R. G. Ingalls, 1375-1386. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Bocciarelli, P., A. D'Ambrogio, A. Giglio and D. Gianni. 2013. "A Saas-Based Automated Framework To Build And Execute Distributed Simulations From Sysml Models." In *Proceedings of the 2013 Winter Simulation Conference*, edited by R. Pasupathy, S.-H. Kim, A. Tolk, R. Hill, and M. E. Kuhl, 1371-1382. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Casalicchio, E., S. Bologna, L. Brasca, S. Buschi, E. Ciapessoni, G. D′ Agostino, V. Fioriti and F. Mohabito (2011). "Interdependency Assessment In The ICT-PS Network: The MIA Project Results." *Critical Information Infrastructures Security*. Springer**:**1-12.

Casalicchio, E. and E. Galli (2008). "Federated Agent Based Modeling And Simulation: An Approach For Complex Critical System Analysis." *Principles of Advanced and Distributed Simulation***:**147.

Delligatti, L. 2014. *SysML Distilled: A Brief Guide to the Systems Modeling Language*. Pearson Education Inc.

Fellbaum, C. 1998. WordNet: An Electronic Lexical Database. Bradford Books.

Feng, C.M., C.H. Hsieh and S.-C. Peng. 2010. "Optimization of Urban Bus Routes Based on Principles of Sustainable Transportation." *Journal of the Eastern Asia Society for Transportation Studies*.

Friedenthal, S., A. Moore and R. Steiner. 2012. *A Practical Guide to SysML – The Systems Modeling Language*. Elsevier Inc.

Hofmann, M. A. 2004. "Challenges of Model Interoperation in Military Simulations." *Simulation* 80:659 – 667.

Huang, E., R. Ramamurthy, and L. F. McGinnis. 2007. "System and Simulation Modeling using SysML." In *Proceedings of the 2007 Winter Simulation Conference*, edited by S. G. Henderson, B. Biller, M.-H. Hsieh, J. Shortle, J. D. Tew, and R. R. Barton, 796-803. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

IEEE 2010a. "IEEE 1516 - Standard for Modeling and Simulation High Level Architecture - Framework and Rules."

IEEE 2010b. "IEEE 1516.1 - Standard for Modeling and Simulation High Level Architecture - Federate Interface Specification."

IEEE 2010c. "IEEE 1516.2 - Standard for Modeling and Simulation High Level Architecture - Object Model Template (OMT) Specification."

IEEE 2003. "IEEE 1516.3 - Recommended Practice for High Level Architecture (HLA) Federation Development and Execution Process (FEDEP)."

Kuhl, F., R. Weatherly and J. Dahmann. 1999. *Creating Computer Simulations – An Introduction to the High Level Architecture*. New Jersey: Prentice Hall.

Lu, Z. 2014. "Managing the Complexity of Sustainable Cities: the Interdependence between Infrastructure System and Socioeconomic Environment." Ph.D. thesis, School of Civil and Environmental Engineering, Georgia Institute of Technology, Atlanta, Georgia.

McGinnis, L., E. Huang, K.S. Kwon and V. Ustun. 2011. "Ontologies and simulation: a practical approach." *Journal of Simulation* 5:190–201.

Ouyang, M. 2014. "Review on Modeling And Simulation Of Interdependent Critical Infrastructure Systems." *Reliability Engineering & System Safety*:43-60.

Rathnam, T. 2004. “Using Ontologies To Support Interoperability In Federated Simulation.” Master’s thesis, Department of Mechanical Engineering, Georgia Institute of Technology, Atlanta, Georgia.

Rathnam, T. and C. Paredis. 2004. “Developing Federation Object Models Using Ontologies.” In *Proceedings of the 2004 Winter Simulation Conference*, edited by R .G. Ingalls, M. D. Rossetti, J. S. Smith, and B. A. Peters, 1054-1062. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Rodriguez, M.A. and M. J. Egenhofer. 2003. “Determining Semantic Similarity Among Entity Classes from Different Ontologies.” *IEEE Transactions On Knowledge And Data Engineering* 15:442-456.

Rouquette, N. and S. Jenkins. “OWL Ontologies and SysML Profiles: Knowledge Representation and Modeling.” Systems and Software Division, Jet Propulsion Laboratory, NASA. Accessed April 8. http://www.omg.org/news/meetings/tc/mn/special-events/ecl/Rouquette.pdf.

Schönherr, O., and O. Rose. 2009. “First Steps Towards A General SysML Model For Discrete Processes In Production Systems.” In *Proceedings of the 2009 Winter Simulation Conference*, edited by A. Dunkin, R. G. Ingalls, E. Yucesan, M. D. Rossetti, R. Hill, and B. Johansson, 1711–1718. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Tessuma, C. W., J. D. Hillb and J. D. Marshall (2014). "Life Cycle Air Quality Impacts Of Conventional And Alternative Light-Duty Transportation In The United States." In *Proceedings of the National Academy of Sciences* 111(52):18490-18495.

Wang, W., A. Tolk and W. Wang. 2009. “The Levels of Conceptual Interoperability Model: Applying Systems Engineering Principles to M&S.” In *Proceedings of the 2009 Spring Simulation Multiconference.*

Wu, Z., H. Wu, W. Li and X. Zhang. 2007. “Extending Distributed Simulation’s Run-Time Infrastructure with Web Services.” In *Proceedings of the IEEE International Conference on Automation and Logistics*:1528-1532.

AUTHOR BIOGRAPHIES

AJITESH JAIN is a graduate research assistant at Georgia Institute of Technology and is working towards his Master’s degree in Computer Science. His research interests include modeling and simulation, machine learning and data mining. His email address is ajiteshJain@gatech.edu.

RICHARD FUJIMOTO is Regents’ Professor in the School of Computational Science and Engineering at the Georgia Institute of Technology. He received a Ph.D. in Computer Science and Electrical Engineering from the University of California-Berkeley. His email address is fujimoto@cc.gatech.edu.

JONGCHAN KIM is a Ph.D. student at Georgia Institute of Technology. His research interests include parallel computation for fluid dynamics and modeling of complex systems. His email address is jkim764@gatech.edu.

MENGMENG LIU is a Master’s student at Georgia Institute of Technology. Her research interests are sustainability assessment of the civil infrastructures. Her email address is mliu301@gatech.edu.

JOHN CRITTENDEN is a Professor and Hightower Chair and Georgia Research Alliance Eminent Scholar in Environmental Technologies in the School of Civil and Environmental Engineering at the Georgia Institute of Technology. He is the Director of the Brook Byers Institute for Sustainable Systems. His email address is john.crittenden@ce.gatech.edu.

ZHONGMING LU is a researcher in the Brook Byers Institute for Sustainable Systems at the Georgia Institute of Technology. His email address is Zhongming.Lu@gatech.edu.