Volume: 14 Issue: II Month of publication: February 2026
DOI: https://doi.org/10.22214/ijraset.2026.77564

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue II Feb 2026- Available at www.ijraset.com

Task Specific Markup Languages for Simulation Modeling

Dean C. Chatfield
Department of Information Technology and Decision Sciences, Old Dominion University

Abstract: A task specific markup language (TSML) is a specialized markup language with a syntax and structure designed for a specific purpose. These languages are typically defined using the XML meta-language. The wide adoption of XML and wide availability of XML tools and technologies, such as parsers, make TSML files portable and allow for open accessible standards, interoperability, and inter-application communication. A number of task specific markup languages have been developed and employed to support many aspects of simulation modeling. In this paper we provide an overview of the three basic categories of task specific markup languages for discrete-event simulation: data management and storage, communication, and modeling. We present these application categories with sub-categories and examples from the literature. We then discuss issues related to developing task specific markup language standards for simulation support and point out several important unresolved issues.
Keywords: Simulation, Markup Languages, XML, Information Standards, Structured Information.

I. INTRODUCTION

Open data standards have become increasingly important in recent years. One of the most widely used forms of open data standards is the development and use of markup languages designed to store, process, and communicate specific information or tasks, otherwise known as "task specific markup languages" (TSMLs). Defining a task-specific markup language allows one to define a class of documents, or file formats, in terms of allowable content, required structure, and proper presentation. Since much of the early use of markup languages was in the document management, processing, and printing industries, markup languages became associated with document layout and presentation. HTML, the most widely used markup language, continued that orientation, because it is essentially a way to describe how hypertext will be presented in a browser, instead of on a printed page. However, it was soon realized that markup languages are well-suited for describing information structure and relationships in addition to its appearance. This has driven attention to the use of markup languages to support modeling methods, including simulation modeling.

Markup language-based documents exhibit great flexibility in storing data of many kinds, from flat-file to relational database to object-oriented models. It is natural for the simulation modeling community to take advantage of these capabilities, and markup languages have been developed and utilized to support simulation modeling in many fashions. We present the landscape of TSML support for simulation, focusing on discrete-event simulation, based on three categories: data management and storage, communications, and modeling. In many cases a TSML-based initiative will crossover to include more than one of the defined purposes. Within each category we describe various sub-categories and sample initiatives and implementations from the literature and industry. We do not attempt to make a comprehensive list of all markup language-based simulation initiatives, but rather seek to define areas where TSMLs can be utilized. We then point out issues and challenges with markup language driven simulation support utilization, and finally draw some conclusions.

II. MARKUP LANGAGE BACKGROUND

A. Markup Languages

A markup language describes the structure or appearance of information contained in a document by the addition of descriptive symbols (markup characters or “tags”) within the document itself and by imposing a structure that the document must follow. Initially, much focus was placed on the use of markup languages for authoring purposes, but it was quickly determined that markup languages are also applicable as methods of information structuring, delivery, and transfer.

A markup language can be one of two basic types: a task-specific markup language (TSML) or a meta-language. A task-specific markup language is a language used for creating a document that conforms to a pre-defined specification regarding content, structure, and other aspects. We note that some use the term "domain specific markup language" instead of "task specific markup language", however we feel that "task specific" is more appropriate as it indicates the markup language is focused on a particular task or activity area, which is our primary interest. The HyperText Markup Language (HTML) is a common example of a task-specific markup language.

A meta-language is used for defining a task-specific markup language, resulting in the specification for a new class, or type, of document. Using a meta-language, one can define the markup symbols and the overall structure of a document type. By defining all the pieces that may exist in a document (syntax), as well as the rules for putting those pieces together to form a valid document, a class of documents is unambiguously specified. Examples of meta-languages are the Standard Generalized Markup Language (SGML) and XML [1], with XML being significantly more popular in recent years.

The description of a new markup language (structure, content, syntax, etc.) using a meta-language is referred to as a *schema* (note the lower-case "s"). With XML, there are multiple approaches to developing a schema, generally referred to as schema languages, including Document Type Definition (DTD), XML Schema Definition (XSD) which is often referred to as "XML Schema", Schematron, and Regular Language for XML Next Generation (RELAX NG). Early markup language development focused on the use of DTDs for language specification, but in more recent years the utilization of other approaches, such as XSD, has grown significantly.

As an example, we find the Geographic Markup Language (GML) to be a task-specific markup language designed to enable the open exchange of descriptions of geographical features. GML was defined using XML, specifically through the use of XML Schema Definition (XSD).

B. Advantages of Using a Task Specific Markup Language

Task specific markup language offer a number of advantages. The most fundamental benefit of a TSML is that a document can then be put through a validation procedure to designate it as truly valid and conforming to the specification. When defining a TSML the structure, content, syntax, and potentially the layout of the document will be specified. Thus, we can enforce a set of "rules" and only allow documents that match the agreed-upon specification. One of the results of this is enhanced data sharing capabilities since the document specification (schema) will enable all parties to know how to create a valid and usable document. In addition, this allows the development of efficient markup document processing routines (for example, specialized parsers) since the documents will have a known structure and content. Thus, software such as specialized TSML-specific editors may be built, and simulation model generators may be possible. Task specific markup languages benefit from being derived from an XML specification due to the mature infrastructure and wide adoption of general XML document processing technologies. Thus, a TSML has a distinct advantage over a proprietary non-markup language data format that will need to have all the associated technologies created specifically for it. Thus, a TSML leverages the developments of its "parent" (XML), while providing additional necessary restrictions on the content and structure of the document type.

III. TASK SPECIFIC MARKUP LANGAUGE (TSML) APPLICATIONS FOR SIMULATION MODELING

We partition the potential applications of TSMLs to assist simulation modeling into three categories: data management and storage, communication, and modeling. For each category we provide a brief explanation of TSML applications, as well as particular sub-areas and examples from the literature.

A. Data Management and Storage

Simulation requires and generates many different forms of information. Open, standardized representations for data, such as user empirical distributions and time series, are valuable for distributed computing, analysis, and output sharing. At a basic level, TSMLs have been used for structuring input and output files where data need to be stored and extracted.

1) Input Data

On the input side, we have various forms of information required by simulation models, including empirical data (arrivals, service times, other distributions), scenario data, and experimental information (runs, replications).

One of the most comprehensive projects to create standard storage formats and interfaces for simulation modeling has been by the U.S. National Institute of Standards and Technology (NIST). NIST has pursued the development of standard, XML-based data file formats (TSMLs) for machine shop simulation [2]. NIST uses UML (Unified Modeling Language) for describing the model of the job shop, and a task specific markup language (TSML) to define the data formats for the information described using UML [3], [4], [5]. McLean *et al.* [3] have also developed an architecture for a machine shop simulation tool, using Rockwell’s Arena simulation environment, that interfaces directly with the TSML data formats.

Kokkonen, Jolma, and Koivusalo [6] describe the utilization of TSML data formats for interfacing input sources in one of the most data-laden simulation domains, environmental modeling. Kokkonen *et al.* [6] present an XML/DBMS-based approach for allowing a simulation model to interface with a relational database containing input, such as meteorological data.
A related use of TSMLs is the Simulation Data exchange (SDX) format, which is designed to be a standardized format for representing AutoCAD layouts of facilities in a text-based, model-friendly format. This has become a supported format of several commercial simulation packages such as AutoMod and Simul8, which provides opportunity for model exchange and even automated model generation [7].

2) Output Data
On the output side, we see the need to store information of many kinds, such as reports, summary (replication) data, time series data, and animation scripts. Utilizing an open, structured format, such as a TSML document, allows ease of output sharing and the opportunity to automate analysis by developing applications that are “aware” of the format of the simulation output. Stein, Bachta, and Baker [8] provide a complete description, including the DTD, of an IOM_output format used to capture output from the Inverse Ocean Modeling (IOM) Simulator.

3) Scenario Data
Simulation projects generally involve testing several different configurations, or “scenarios” of a system. Storing scenario descriptions in TSML documents allows for better experiment management, easier sharing of scenario data, and the possibility of auto-generating the appropriate set of scenario data directly from an experimental design software. The use of a standard format to describe and store simulation scenarios has been examined by Hout [9], who discusses the use of an XML-based format to represent Naval Simulation Systems (NSS). Hallagan [10] describes a pairing of model and experiment languages created for network simulation modeling while Asasdi and Schubert [11] discuss the development of structured document format for storing scenario data used for simulating military operational plans. Wilsdorf *et al.* [12] describe the development of schema-based document formats for more generalized, yet detailed, simulation scenario storage.

4) Animation
Complex animations have become a mainstay of modern simulation software. The ability to “outsource” the animation to a possibly external engine, driven by (XML-based) animation scripts describing what occurred in a simulation run, has attracted attention. In addition, animation control has been aided by the new x3D standard, which extends the older virtual reality markup language (VRML). The x3D language is a TSML for defining 3D animations. Simulation tool builders have begun to incorporate mechanisms for generating x3D files from simulation runs as a means of creating portable, high quality 3D animations. The XMSF project [13] and RUBE [14], [15], [16] have added 2D and x3D capabilities for animation and model visualization.

5) Model Storage
TSMLs can be used as a simulation tool’s model storage file format, an example being that used by CACI’s SimProcess tool. The use of an XML-based TSML file format “internally” within a simulation software package provides the benefit of standardized data access, making it easier to add features and enabling cross-product integration. Chen and Chen [17] utilize this approach in their supply chain simulation system while Touraille *et al.* [18] adopt this as part of their proposed standard for linking DEVS and non-DEVS systems and Hallagan [10], as noted above, develop a pair of model and scenario storage TSMLs.

B. Communication
Many new forms of simulation models use some components and data that are not “local”, i.e., not at the same location as the modeler. In addition, the ability to move standardized information is important for multi-modeling, hierarchical models, distributed models, federates, and web-based simulation models. To make these systems operate effectively, a means of communication is necessary, and TSML document formats provide some of the necessary capabilities. XML is not a telecommunications standard or protocol, but the definition of standardized formats (TSMLs) for information interchange is just as important to communication as the actual telecommunication aspects of moving information.
Several of the formats and languages designed for storage of input data are also utilized for communication. The NIST data standard described in the previous section is an enabler of communication, even though it is primarily concerned with permanent data storage as opposed to temporary storage for communication purposes.

1) Interoperability and Multi-modeling

The concept of interoperability involves the utilization of multiple platforms or simulation tools which require data to be passed between them in order to create a useful, functioning simulation system from distributed components. Interoperability is critical to distributed simulation systems as well as simpler, but multi-platform, systems. The practice of multi-modeling may involve supporting more than one model instance, having more than one model type, having a hierarchy of model instances, or alternate presentations of the same model. To support these situations, a method of coupling the various platforms, tools, models, or instances is needed. Fishwick [19] presents the multi-modeling exchange language (MXL) as a means of supporting multi-modeling. Part of the HDPS system of Curry and Vlahos [20] includes communication formats to support hierarchical modeling. Mittal *et al.* [21] propose the DEVSML language to provide interoperability between DEVS models distributed across multiple locations, communicating via web services. Harward and Harrell [22] discuss using the NIST Shop Data Model (SDM) as a foundation for a general format for simulation data exchange and interfacing. Touraille *et al.* [18] explicitly define interoperability as an important part of their proposed standard for linking DEVS and non-DEVS systems. Rioux *et al.* [23] describe an interesting approach utilizing TSMLs to provide a "technology-unified data pipeline targeted to interactive simulation".

2) Distributed and Web-based Simulation

Web-based simulation and simulation web services are method for delivering simulation capabilities via the internet. Distributed simulation is the execution of simulations on several computers connected via a network, usually the internet. Although different, each of these simulation technologies require a means of communicating of model components over the internet. TSMLs defined via XML provides a logical choice for communication requirements because several communications-based web technologies and standards are XML-based, such as Simple Object Access Protocol (SOAP) [24] , or have the option to utilize XML-based standards, as seen in Representational State Transfer (REST).

One of the most ambitious endeavors in this area has been the extensible Modeling and Simulation Framework (XMSF). XMSF is a set of web-based formats and technologies for modeling and simulation projects, developed by a group of U.S. military institutes and associated research organizations [13]. The idea of XMSF is to create a flexible, extensible framework, based on web technologies and open standards, to support development of a new generation of modeling and simulation applications, while continuing support for current modeling and simulation technologies. XMSF utilizes XML-defined TSMLs for a majority of its model storage, data storage, and communication needs, but makes its biggest impact as an overall set of guidelines for developing web and XML-enabled simulation applications. Additionally, Bergmann *et al.* [25] describe a method for distributing discrete-even simulation experiment information via web-services technologies, utilizing a TSML based on the Core Manufacturing Simulation Data Model (CMSD) standard created by NIST [22] and others.

C. Modeling

Our third category of TSML applications for simulation modeling is the use of these document formats in the model building process itself. These involve XML-based documents and related technologies that guide, drive, or automate the modeling process. In some cases, the modeling documents and technologies are included as part of a broader framework that includes storage and communication standards as well.

1) TSML-Based Simulation Languages and Model Specifications

TSMLs can be used to define a simulation model. One approach is to create a simulation language based on XML constructs. Curry and Vlahos [20] describe a modeling system called the Hierarchical Dynamic multi-Paradigm System (HDPS), designed to model business and financial systems composed of several interacting sub-systems. HDPS separates a simulation model into four parts: type, definition, instance, and implementation. Through the use of XML Schema, XML namespaces, and XSLT (extensible Stylesheet Language Translations), HDPS uses a TSML to specify types, definitions, and instances. This amounts to the entire model, with the exception of the code for a particular simulation platform. A .Net implementation of the HDPS concept, called 'xHDPS,' has been constructed.

Wiedemann [26] proposes an TSML-based, language-independent standard for representing Simulation Modeling Language (SML) [27], [28], [24] statements, called 'OpenSML.' Code converters are used to generate language-specific statements from the generic XML version when the model is to be executed. Models built with OpenSML are transferred into executable Java, .Net, C++, and C#-based simulation models via document parsing tools. In addition, Meseth [29] presents an approach to defining a simulation model using a TSML named XLSC (created via an XML Schema) which can then be interpreted to allow execution of the model.

A number of projects have focused on using TSMLs for simulation model specification. While not full simulation languages, these specification formats provide a number of benefits for simulation modeling, depending on the situation and system being simulated. Rohl and Uhrmacher [30] describe an XML data-binding approach to defining simulation models while Rohl and Morgenstern [31] describe an approach using an adaptation of the Web Service Description Language (WDSL). Olson et al. [32] describe CS-XML, a TSML that is part of a three-part system for defining, translating, and executing simulation models. Schonherr and Rose [33] describe an approach to create a general implementation of the System Modeling Language (SysML) that includes model specification as well as simulation model generation. Zhang *et al.* [34] describe the utilization of a TSML for supply chain simulation models based on the Supply Chain Operations Reference model (SCOR). Liehr and Buchenrieder [35] create a TSML based on an XML Schema for simulating extended queuing networks. Likewise, Troya and Vallecillo [36] present xQNM, a TSML for defining models of extended queuing networks. Hu *et al.* [37] describe their DEVSMO ontology (based on the OWL language) which allows for XML based model representations. Alshareef *et al.* [38] present a DEVS simulation system which utilizes a TSML named DNL (DEVS Natural Language) to define the simulation system, which can then be translated to a Java based simulation for execution.

2) *TSML Intermediary File*

A second approach to simulation modeling support is to use a markup language-based intermediary file. Wang and Lu [39] discuss a tool that allows the user to describe the model they wish to create, which is then translated into an intermediate TSML document format. The model specification can then be converted into discrete event simulation code for various environments via translation modules. This approach allows one to store the model in a platform-independent format. Lacy and Gerber [40] describe the Web Ontology Language (OWL), which can be used for deep semantic descriptions of a system, which can then be translated into a simulation tool or language. Gyimesi [41] presents the web services focused SimASP framework, which utilizes a task specific markup language as an intermediary format for storing and communicating model generic DES model information. Chen and Chen [17] utilize a TSML file format as an intermediary in their supply chain simulation system.

3) *"Executable" TSML Model File*

A third approach is used by Reichenthal [42] in the development of the Simulation Reference Markup Language (SRML). SRML integrates XML data models with behavior descriptions defined as scripts. This combination of descriptive information plus executable scripts makes an SRML model self-contained. All that is needed is software to interpret the contents and execute the code in the correct manner, a function performed by the Simulation Reference Simulator (SR Simulator) [43]. SRML was submitted to the World Wide Web Consortium (W3C) and is available to the public. Meseth [44], [29] specifies the development of XSLC, a TMSL that can be interpreted, thus allowing for model execution. Dahmani *et al.* [45] specify a TSML for defining a DEVS simulation, which can then be executed via a set of XSLT transformations.

4) *TSML Simulation "Building Blocks"*

A task specific markup language can be used to create a set of simulation building-blocks, from which models can be constructed. This idea is used by Gustavson and Chase [46], who utilize an XML-defined TSML in the definition of Base Object Models (BOMs). BOMs capture a set of activities among entities and package them as reusable “building blocks”. Their idea is that BOMs and SRML [47] can be used within the XMSF framework, previously described, to allow users to rapidly develop simulation models from certified building blocks. Rohl and Uhrmacher [48] describe composing simulations from a set of components defined in an XML Schema.

5) *Simulation Model Generators*

Task specific markup languages, couple with their associated XML technologies, have also been utilized to *generate* simulation models automatically, based on a description in a TSML document containing the format of the system to be simulated. This approach is fundamentally different from the simulation language, storage format, or building-block approaches described above. The model generating approach typically relies on XML processing technologies, such as DOM parsers, SAX parsers, and XSLT to translate a system description, stored as a TSML file, into a working simulation model for a particular simulation platform. A general simulation model generator requires a user to describe the problem in simulation term, but can be utilized for various problem domains, whereas a domain specific simulator allows the user to use the language of the subject matter, at the cost of restricting the use of the simulator to a particular domain.

A general simulation model generator was presented by Arief and Speirs [49]. Their approach allows the user to define the system they wish to simulate in terms of simulation primitives, such as queues, objects, and data, utilizing an UML-based sequence diagram. The resulting description is stored as a SimML (Simulation Modeling Language) file. A Java application is used to parse the SimML file utilizing a Simple API for XML (SAX) parser. The parser is customized so that it reads through the SimML file and generates an appropriate model for the JavaSim simulation environment.

Chatfield, Harrison, and Hayya [50], [51] describe a domain-specific Simulator for Integrated Supply Chain Operations (SISCO). SISCO allows users to depict a supply chain's structure, operations, and behavior, and store that description as an XML-based Supply Chain Modeling Language (SCML) file [52]. The SISCO Engine contains a customized SAX parser that reads the SCML file and creates an equivalent simulation model for the SILK simulation environment by utilizing a set of supply chain simulation building-block classes. The result is an object-oriented simulation model that represents the supply chain described in the SCML file.

Canoncio, Donato, and Ventre [53] present a markup language designed to represent a network system. The Network Simulation Description Language (NSDL) is a document format, defined as an XML schema, which contains a structural and behavioral description of a network in a standardized format. NSDL files can be processed with the XSLT transformation language to translate, via a set of custom translation rules, the contents of the NSDL file into a simulation script for the popular ns-2 network simulator. Schonherr and Rose [33] include model generation capabilities in their generalized SysML implementation, as noted above. Sprock and McGinnis [54] describe task specific markup language usage in their model-to-model transformation capable system for logistics oriented DEVS models. Finally, Dahmani *et al.* [45], as noted previously, utilize XSLT transformations to generate a DEVS Simulation from a TSML based system definition.

IV. DISCUSSION: IMPORTANT FACTORS AND UNRESOLVED ISSUES

As presented above, the use of task specific markup languages in simulation modeling comes in many forms. When deciding on a potential XML application, there are factors that must be taken into account. In addition, there are several issues that impact the overall effectiveness of markup formats for simulation support. We present these based on five categories: scope, modeling issues, technical issues, organizational issues, and competition.

A. Scope

When embarking on a project to develop a task specific markup language to support simulation modeling, the first concern must be to precisely define what it is that will be standardized with the TSML. In previous sections, we saw data management, communications, and modeling uses of task specific markup, with several varieties of applications within each category. Properly defining the scope of the document format is vital. In addition to the application area, accurately defining the scope of an markup document format for simulation includes aspects such as the level of modeling resolution, the desire for extensibility, and the scalability of the standard. It has become increasingly popular to include simulation as part of multi-tool analysis "workbenches," which are generally domain-specific. Optimization and other methodologies require different types of data and, of course, represent the system differently than simulation does. Thus, whether to include non-simulation data in the markup document is important and leads to several unresolved issues regarding model storage, data storage, output, and other possible areas for markup language-based support.

B. Modeling Issues

Modeling issues can have a profound impact on the design of markup formats for simulation, especially those used to support model building. The first is the recognition that different modeling types will require inherently different information. Standard discrete event models, object-oriented simulation models, and agent-based simulation models, for example, differ in the information that is required, and the markup document file format must account for this. Adding to the complexity are the different "world-views" that can be employed for any particular approach. This variety of approaches makes the development of a "universal" markup language simulation model format impractical. We believe a more attainable goal is to identify modeling styles or domains and create appropriate standards for each.

Modeling style variety leads to another issue: whether the TSML formats should be detailed standards or general frameworks for simulation modeling. Generally, markup language documents have been employed to represent specific information using fairly detailed formatting specifications. However, as in the case of XMSF [13], for example, markup languages can be utilized to create general frameworks or standards.

These are flexible, and thus applicable to a wider variety of situations, but at the risk of incomplete standardization and interoperability gaps. If the goal is to utilize a task specific markup language as part of a model generating system, then detailed standards are necessary in order to adequately formulate the set of rules needed to create a working simulation model from a markup language-based system description.

Due to its wide variety of approaches, world-views, and application domains, simulation modeling, is more difficult to develop standard data and model formats for as compared to the more structured methodologies, like mathematical programming. For example, the field of mathematical programming has many standard formulations (cutting stock problems, mixture problems, transportation problems) applicable to a wide variety of situations.

Simulation does not have such fundamental models, other than the overly simplistic single server-single queue, to use as a foundation for a set of model descriptions and a format to support them.

In almost any description of the life-cycle of a simulation project, for example Shannon [55], we see that model building ("coding") takes only a small portion of the allotted time. Data collection and conceptual model formulation are allotted large amounts of time and should be considered for standardized formats and markup-based assistance. A commonly overlooked part of the life-cycle that is extremely important to simulation modeling projects is model verification and validation. This issue has received minimal attention from the markup language community, and it should be deserving of some investigation.

C. Technology Issues

Task specific markup languages are defined almost exclusively using XML and utilize XML related technologies for processing and management. Issues regarding the technical capabilities of the underlying XML technologies are important, and not completely resolved. First, XML is not completely object-oriented, as it lacks some fundamental object-oriented capabilities, such as inheritance. Since the use of object-oriented modeling is quite common, this poses a problem for storing an object-oriented model or problem (system) description. The XML Schema Definition (XSD) schema language adds some object-oriented features, several extensions exist, such as SOX (Schema for Object-oriented XML) to add OO features, plus the judicious use of XML “entities” can be employed to enable XML to attain object-oriented capabilities. However, since different developers may employ different extensions and approaches, the access to documents using these may not be universal.

Second, XML and, by extension, TSML files consist of plain text, i.e., standard ASCII characters, which causes problems. First, XML is, in addition to being a plain text format, a relatively verbose language, making XML files somewhat cumbersome to read, and also large in size. The inflated size means that sending XML files over a network will require additional bandwidth. Fortunately, plain text files are readily compressible with open compression-decompression standards. However, the same compression-decompression standards need to be adopted by all simulation participants in order for compression to be a solution to the problem of large file size.

Third, XML files have minimal security features, such as encryption or access validation. Thus, data, model, or communication files may be captured by others, compromising sensitive data or results. An additional concern is the potential use of XML and related markup documents as a vector of system attack through malformed documents. This is an issue that must be addressed for XML and TSML languages to have a role in important simulation areas, such as national defense, terrorism modeling, and network security analysis.

D. Organizational Issues

Some issues facing the utilization of markup language document formats for simulation modeling are related to the organization or business in which the modeling will occur. When deciding on a standard, the cost of supporting the standard may be an issue. Development costs, support costs, integration costs, training costs, and the like need to be balanced against the perceived benefits.

Another problem is that the XML meta-language is so flexible that standards coordination is a major issue. XML makes it relatively easy to define new, specialized file formats for information storage, modeling, and communication. The upside is that a TSML format that exactly matches the needs can be developed. The downside is that if every tool or project results in unique formats being developed, resulting in over-fragmentation of the solution developments. If this is the case, then there would be no true standardization and the goals of reuse, interoperability, and communication will be hampered.

Finally, we must be concerned with whether the information will be used solely for simulation modeling or whether it will have other purposes. This could have impact on the format’s design or on the need to support another kind of application.

E. Competition

The most recent years have seen a deceleration in new TSML developments for simulation support. Some of this deceleration is undoubtedly the result of a matured landscape of previously developed markup document types, but we also note that task specific markup languages for simulation modeling support face competition on two fronts. First is competition on the technology front, namely the rapid ascent of JSON and similar data transfer languages, such as YAML. These languages are not specifically tailored to an application area or task (as TSMLs are), but are a generic structure that can be used for data storage and communication. However, these languages are simple, have low overhead, and are readily available as the capability to utilize JSON or YAML is becoming a standard feature of many development tools. Thus, some of the simulation support language "marketplace" is choosing simplicity over structure. The second front of competition that TSMLs face is the "mindspace" front. Since the basic concept of markup languages is relatively mature, newer technologies will attract some attention from researchers and industry developers that would have previously been focused on TSML creation. The fact that markup languages are a well-developed area is not a negative development, as it is recognized that specialized markup languages have definite advantages and serve a useful purpose, but it will take additional effort to promote the benefits than has been required in the past.

V. CONCLUSIONS

Task specific markup languages, primarily based on XML definitions, have been employed to support many aspects of simulation modeling. In this paper, we describe the three basic categories of XML application for discrete event simulation modeling: data management, communication, and modeling, with sub-categories and examples from the literature provided for each category. The flexibility of XML and XML-defined file formats (TSMLs) is a great advantage, allowing modelers to create custom data and file structures that match their needs precisely. The large-scale adoption of XML and wide availability of XML tools and technologies, such as parsers, make task specific markup language files portable and allows for open accessible standards, interoperability, and inter-application communication. However, there are issues related to scope, simulation modeling decisions, XML technology, organizational issues, and competition that must be attended to in order to realize the maximum benefit of task specific markup language support for simulation modeling.

REFERENCES

[1] R. Jelliffe, The XML and SGML Cookbook: Recipes for Structured Information, Upper Saddle River, NJ: Prentice Hall, 1998.
[2] Y. Lee, C. McLean and G. Shao, "A neutral information model for simulating machine shop operations," in Proceedings of the 2003 Winter Simulation Conference, Chick, S., Sanchez, P.J., Ferrin, D., & Morrice, D.J., eds, Piscataway, New Jersey, 2003.
[3] C. McLean, C. Jones, T. Lee and F. Riddick, "An architecture for a generic data-driven machine shop simulator," in Proceedings of the 2002 Winter Simulation Conference, Yucesan, E., Chen, C-H., Snowdon, J.L. & Charnes, J.M., eds., Piscataway, New Jersey, 2002.
[4] R. Lu, G. Qiao and C. McLean, "NIST XML Simulation Interface Specification at Boeing: A Case Study," in Proceedings of the 2003 Winter Simulation Conference, Chick, S., Sanchez, P.J., Ferrin, D., & Morrice, D.J., eds., Piscataway, New Jersey, 2003.
[5] Y. Lee and Y. Luo, "Data Exchange For Machine Shop Simulation," in Proceedings of the 2005 Winter Simulation Conference M. E. Kuhl, N. M. Steiger, F. B. Armstrong, and J. A. Joines, eds., Piscataway, New Jersey, 2005.
[6] T. Kokkonen, A. Jolma and H. Koivusalo, "Interfacing environmental simulation models and databases using XML," Environmental Modeling and Software, vol. 18, no. 5, pp. 463-471, 2003.
[7] D. Sly and S. Moorthy, "Simulation data exchange (SDX) implementations and use," in Proceedings of the 2001 Winter Simulation Conference, Peters, B.A., Smith, J.S., Medeiros, D.J., & Rohrer, M.W., eds, Piscataway, New Jersey, 2001.
[8] R. Stein, E. Bachta and M. Baker, "An XML structure for IOM simulation output," Visualization and Interactive Spaces Lab, Pervasive Tech Labs at Indiana University, 2003.
[9] G. Hout, "Toward XML Representation of NSS Simulation Scenario for Mission Scenario Exchange Capability," Master's Thesis, MOVES curriculum, Naval Postgraduate School, Monterey CA, 2003.
[10] A. Hallangan, "The Design of XML-Based Model and Experiment Description Lanaguages for Network Simulation," 2011. [Online]. Available: https://digitalcommons.bucknell.edu/honors_theses/43/.
[11] H. Asadi and J. Schubert, "A Stochastic Discrete Event Simulator for Effects-Based Planning," in Proceedings of the 2013 Winter Simulation Conference, Pasupathy, R., Kim, S.-H., Tolk, A., Hill, R., & Kuhl, M. E., eds, Piscataway, New Jersey, 2013.
[12] P. Wilsdorf, M. Dombrowsky, A. Uhrmacher, J. Zimmermann and U. van Rienen, "Simulation Experiment Schemas – Beyond Tools And Simulation Approaches," in Proceedings of the 2019 Winter Simulation Conference N. Mustafee, K.-H.G. Bae, S. Lazarova-Molnar, M. Rabe, C. Szabo, P. Haas, and Y.-J. Son, eds., Piscataway, New Jersey, 2019.
[13] C. Blais, D. Brutzman, D. Drake, D. Moen and K. Morse, "Extensible Modeling and Simulation Framework (XMSF) 2004 Project Summary Report," 2004. [Online]. Available: https://apps.dtic.mil/sti/citations/ADA431159.
[14] P. Fishwick, L. Jinho, P. Minho and S. Hyunju, "RUBE: A customized 2D and 3D modeling framework for simulation," in Proceedings of the 2003 Winter Simulation Conference, Chick, S., Sanchez, P.J., Ferrin, D., & Morrice, D.J., eds., Piscataway, New Jersey, 2003.
[15] H. Shim and P. Fishwick, "Programming Using Dynamic System Modeling Via A 3D-Based Multimodeling Framework," in Proceedings of the 2005 Winter Simulation Conference M. E. Kuhl, N. M. Steiger, F. B. Armstrong, and J. A. Joines, eds., Piscataway, New Jersey, 2005.

[16] H. Shim, "An Xml-Based Diagrammatic Dynamic Modeling And Simulation System," University of Florida, 2003.
[17] Y.-J. Chen and Y.-M. Chen, "An XML-Based Modular System Analysis and Design for Supply Chain Simulation," Robotics and Computer-Integrated Manufacturing, vol. 25, pp. 289-302, 2009.
[18] L. Touraille, M. Traore and D. Hill, "A Mark-up Language for the Storage, Retrieval, Sharing and Interoperability of DEVS Models," in SpringSim '09: Proceedings of the 2009 Spring Simulation Multiconference, SanDiego, CA, 2009.
[19] P. Fishwick, "Using XML for Simulation Modeling," in Proceedings of the 2002 Winter Simulation Conference, Yucesan, E., Chen, C-H., Snowdon, J.L. & Charnes, J.M., eds., Piscataway, New Jersey:, 2002.
[20] R. Curry and K. Vlahos, "HDPS: An XML/XSLT based hierarchical modeling system," in Proceedings of the 2004 Winter Simulation Conference, Ingalls, R.G., M.D. Rossetti, J.S. Smith, and B.A. Peters, eds., Piscataway, New Jersey, 2004.
[21] S. Mittal, J. Risco-Martín and B. Zeigler, "DEVSML: Automating DEVS Execution Over SOA Towards Transparent Simulators," in SpringSim 2007: Proceedings of the 2007 Spring Simulation Multiconference, San Diego, CA, 2007.
[22] G. Harward and C. Harrell, "Assessment if the NIST Shop Data Model as a Neutral File Format," in Proceedings of the 2006 Winter Simulation Conference, L. F. Perrone, F. P. Wieland, J. Liu, B. G. Lawson, D. M. Nicol, and R. M. Fujimoto, eds., Piscataway, New Jersey, 2006.
[23] F. Rioux, F. Bernier and D. Laurendeau, "Design and Implementation of an XML-Based, Technology-Unified Data Pipeline for Interactive Simulation," in Proceedings of the 2008 Winter Simulation Conference, S. J. Mason, R. R. Hill, L. Mönch, O. Rose, T. Jefferson, J. W. Fowler eds., Piscataway, New Jersey, 2008.
[24] R. Kilgore, "Simulation Web Services With .NET Technologies," in Proceedings of the 2002 Winter Simulation Conference, Yucesan, E., Chen, C-H., Snowdon, J.L. & Charnes, J.M., eds., Piscataway, New Jersey, 2002.
[25] S. Bergmann, S. Stelzer, S. Wustemann and S. Strassburger, "Model Generation in SLX Using CMSD and XML Stylesheet Transformations," in Proceedings of the 2012 Winter Simulation Conference, C. Laroque, J. Himmelspach, R. Pasupathy, O. Rose, and A.M. Uhrmacher, eds., Piscataway, New Jersey, 2012.
[26] T. Weidemann, "Next Generation Simulation Environments Founded on Open Source Software and XML-Based Standard Interfaces," in Proceedings of the 2002 Winter Simulation Conference, Yucesan, E., Chen, C-H., Snowdon, J.L., & Charnes, J.M., eds., Piscataway, New Jersey, 2002.
[27] R. Kilgore, "Open Source SML and SILK for Java-based, object-oriented simulation," in Proceedings of the 2001 Winter Simulation Conference, Peters, B.A., Smith, J.S., Medeiros, D.J., & Rohrer, M.W., eds., Piscataway, New Jersey, 2001.
[28] R. Kilgore, "Open Source Simulation Modeling Language (SML)," in Proceedings of the 2001 Winter Simulation Conference, Peters, B.A., Smith, J.S., Medeiros, D.J., & Rohrer, M.W., eds., Piscataway, New Jersey, 2001.
[29] N. Meseth, "XML-based DEVS Modeling and Interpretation," University of Osnabrueck, 2011.
[30] M. Rohl and A. Uhrmacher, "Flexible Integration of XML Into Modeling and Simulation Systems," in Proceedings of the 2005 Winter Simulation Conference, M.E. Kuhl, N. M. Steiger, F. B. Armstrong, and J. A. Joines, eds, Piscataway, New Jersey, 2005.
[31] M. Rohl and S. Morgenstern, "Composing Simulation Models Using Interface Definitions Based On Web Service Descriptions," in Proceedings of the 2007 Winter Simulation Conference S. G. Henderson, B. Biller, M.-H. Hsieh, J. Shortle, J. D. Tew, and R. R. Barton, eds., Piscataway, New Jersey, 2007.
[32] K. Olson, C. Overstreet and E. Derrick, "Code Analysis and CS-XML," in Proceedings of the 2007 Winter Simulation Conference, S. G. Henderson, B. Biller, M.-H. Hsieh, J. Shortle, J. D. Tew, and R. R. Barton, eds., Piscataway, New Jersey, 2007.
[33] O. Schönherr and O. Rose, "First Steps Towards A General Sysml Model For Discrete Processes In Production Systems," in Proceedings of the 2009 Winter Simulation Conference M. D. Rossetti, R. R. Hill, B. Johansson, A. Dunkin and R. G. Ingalls, eds., Piscataway, New Jersey, 2009.
[34] X. Zhang, Y. Duan, C. Hu and J. Wang, "Process-based Supply Chain Resources Descriptive Model and Knowledge Representation Based on XML," in 2007 International Conference on Wireless Communications, Networking and Mobile Computing, Piscataway, New Jersey, 2007.
[35] A. Liehr and K. Buchenrieder, "An XML Based Simulation Method for Extended Queuing Networks," in Proceedings 22nd European Conference on Modelling and Simulation, 2008.
[36] J. Troya and A. Vallecillo, "Specification and simulation of queuing network models using Domain-Specific Languages," Computer Standards & Interfaces, vol. 36, pp. 863-879, 2014.
[37] Y. Hu, J. Xiao, H. Zhao and G. Rong, "DEVSMO: an ontology of DEVS model representation for model reuse," in Proceedings of the 2013 Winter Simulation Conference R. Pasupathy, S.-H. Kim, A. Tolk, R. Hill, and M. E. Kuhl, eds, Piscataway, New Jersey, 2013.
[38] A. Alshareef, C. Seo, A. Kim and B. Zeigler, "Devs Markov Modeling And Simulation Of Activity-Based Models For MBSE Application," in Proceedings of the 2021 Winter Simulation Conference S. Kim, B. Feng, K. Smith, S. Masoud, Z. Zheng, C. Szabo, and M. Loper, eds., Piscataway, New Jersey, 2021.
[39] Y.-H. Wang and Y.-C. Lu, "An XML-based DEVS modeling tool to enhance simulation interoperability," in Proceedings of the 14th European Simulation Symposium, A. Verbraeck, W. Krug, eds., Dresden, Germany, 2002.
[40] L. Lacy and W. Gerber, "Potential Modeling And Simulation Applications Of The Web Ontology Language - Owl," in Proceedings of the 2004 Winter Simulation Conference R .G. Ingalls, M. D. Rossetti, J. S. Smith, and B. A. Peters, eds., Piscataway, New Jersey, 2004.
[41] M. Gyimesi, "Web Services with Generic Simulation Models for Discrete Event Simulation," Mathematics and Computers in Simulation, vol. 79, pp. 964-971, 2008.
[42] S. Reichenthal, "SRML- Simulation Reference Markup Language," 2002. [Online]. Available: http://www.w3c.org/TR/SRML/.
[43] S. Reichenthal, "SRML case study: simple self-describing process modeling and simulation," in Proceedings of the 2004 Winter Simulation Conference, Ingalls, R.G., M.D. Rossetti, J.S. Smith, and B.A. Peters, eds, Piscataway, New Jersey, 2004.
[44] N. Meseth, P. Kirchhof and T. Witte, "Xml-Based Devs Modeling And Interpretation," in SpringSim '09: Proceedings of the 2009 Spring Simulation Multiconference, San Diego, CA, 2009.
[45] Y. Dahmani, H. Ali and A. Boubekeur, "XML-based DEVS Modelling and Simulation Tracking," International Journal of Simulation and Process Modelling, vol. 15, no. 1/2, pp. 155-169, 2020.
[46] P. Gustavson and T. Chase, "Using XML and BOMs to rapidly compose simulations and simulation environments," in Proceedings of the 2004 Winter Simulation Conference, Ingalls, R.G., M.D. Rossetti, J.S. Smith, and B.A. Peters, eds., Piscataway, New Jersey, 2004.
[47] S. Reichenthal, "Re-Introducing Web-Based Simulation," in Proceedings of the 2002 Winter Simulation Conference, Yucesan, E., Chen, C-H., Snowdon, J.L., & Charnes, J.M., eds., Piscataway, New Jersey, 2002.

[48] M. Rohl and A. Uhrmacher, "Composing Simulations From XML-Specified Model Components," in Proceedings of the 2006 Winter Simulation Conference, L. F. Perrone, F. P. Wieland, J. Liu, B. G. Lawson, D. M. Nicol, and R. M. Fujimoto, eds., Piscataway, New Jersey, 2006.
[49] L. Arief and N. Speirs, "Automatic generation of distributed system simulations from UML," in Proceedings of the 13th European Simulation Multi-conference (ESM '99), Warsaw, Poland, 1999.
[50] D. Chatfield, T. Harrison and J. Hayya, "SISCO: A Supply Chain Simulation Tool Utilizing SILK and XML," in Proceedings of the 2001 Winter Simulation Conference, Peters, B.A., Smith, J.S., Medeiros, D.J, & Rohrer, M.W., eds, Piscataway, New Jersey, 2001.
[51] D. Chatfield, T. Harrison and J. Hayya, "SISCO: An Object-Oriented Supply Chain Simulation Tool," Decision Support Systems, vol. 42, no. 1, pp. 422-434, 2006.
[52] D. Chatfield, T. Harrison and J. Hayya, "The Supply Chain Modeling Language (SCML)," Working Paper, Department of Supply Chain and Information Systems, Pennsylvania State University, 2003.
[53] R. Canonico, E. Donato and G. Ventre, "An XML based network simulation description language," in Proceedings of the European Simulation and Modeling Conference, 2003.
[54] T. Sprock and L. McGinnis, "Simulation Model Generation Of Discrete Event Logistics Systems (Dels) Using Software Design Patterns," in Proceedings of the 2014 Winter Simulation Conference A. Tolk, S. Y. Diallo, I. O. Ryzhov, L. Yilmaz, S. Buckley, and J. A. Miller, eds., Piscataway, New Jersey, 2014.
[55] R. Shannon, "Introduction to the art and science of simulation," in Proceedings of the 1998 Winter Simulation Conference, Medeiros, D.J., E.F Watson, J.S. Carson, and M.S. Manivannan, eds, Piscataway, New Jersey, 1998.