International Journal of Modeling, Simulation,
and Scientific Computing
Vol. 4, No. 3 (2013) 1340006 (11 pages)

DOI: 10.1142/S1793962313400060

World Scientific
www.worldscientific.com

SYSTEM ENTITY STRUCTURES FOR SUITES OF SIMULATION MODELS

BERNARD P. ZEIGLER*, CHUNGMAN SEO† and DOOHWAN KIM‡

RTSync Corp. 530 Bartow Drive Suite A
Sierra Vista, AZ, 85635, USA
**zeigler@rtsync.com*
†*cseo@rtsync.com*
‡*dhkim@rtsync.com*

Accepted 13 May 2013
Published 2 July 2013

We describe how to develop a suite of models in the MS4 Modeling Environment (MS4 Me). The approach employs the operation of merging of System Entity Structures supported by the environment. After construction, the suite of models can be hosted on Model Store, the cloud-based repository of models provided by MS4 systems as a basis for further collaborative model development. A suite of models, relating to Health Care is used as an example. In this paper, we review basic concepts of the SES needed to support of suites of simulation models. We then consider the concept of multiple aspects that provides more advanced capabilities to construct and manipulate suites of models. With this background, we go on to discuss a methodology for developing suites of simulation models and cloud-based technology for storing and sharing such models in a marketplace of models. Finally, we discuss future research and developments needed to bring the marketplace into common use by modeling and simulation practitioners.

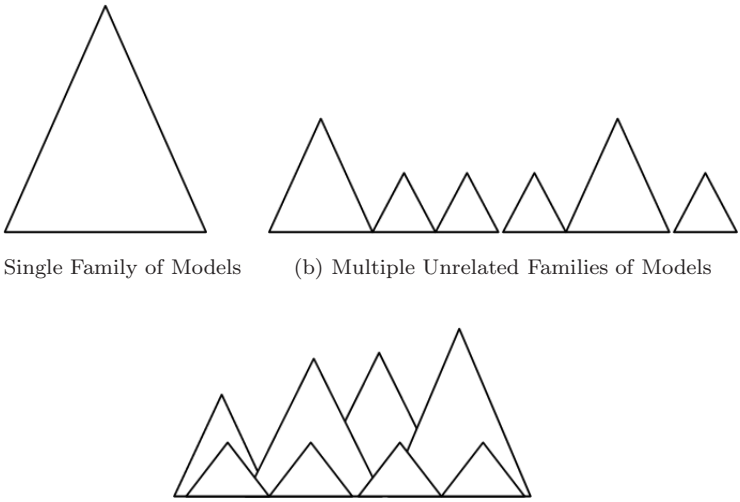
Keywords: System entity structure; suite of models; component-based modeling; systems of systems.

AMS Subject Classification: 68U20

1. Introduction

As has been described in a number of publications (see e.g., Refs. 1–5) the system entity structure (SES) supports development, pruning, and generation of a family of simulation models. Recent papers introduced a new integrated modeling and simulation environment, MS4 Me, developed by MS4Systems.[5,6,a] An expanded concept of the SES that involves multiple families of models and supported by MS4 Me was introduced in Ref. 7. Figure 1 contrasts a suite of models with a single family of models. In Fig. 1(a), an SES implements a single family of models while in Fig. 1(b) a set of nonoverlapping SESs represents a set of families of unrelated

[a]http://www.ms4systems.com/pages/main.php



(a) Single Family of Models (b) Multiple Unrelated Families of Models

(c) Multiple Related Families of Models

Fig. 1. Suites of Models supported by the SES.

models. Finally, Fig. 1(c) depicts a set of intersecting SESs representing a suite of related families of models. In such a suite, there are SESs that are "components" of other SESs. This use of the term "components" transfers the "component" concept from its use in component-based model construction[8] to the domain of SES construction. Thus, an SES is a component of another SES in the sense that the models the first SES generates are components of models generated by the second SES. The operation of composing DEVS models to create a coupled model[9] is mirrored by the merging operation for composing SESs.[10,11] As will be explained, merging generalizes the DEVS construction process in which individual models can be developed, tested and then composed to create hierarchical models in stage-wise fashion. This is to say, merging supports hierarchical composition in which families of models are generated and tested via pruning and transforming their SESs in bottom-up manner.

In the rest of this paper, we first review some basic concepts of the SES needed to support the suites of simulation models. We then consider the concept of multiple aspects that provides more advanced capabilities to construct and manipulate suites of models. With this background, we go on to discuss a methodology for developing suites of simulation models and cloud-based technology for storing and sharing such models in a marketplace of models. Finally, we discuss future research and developments needed to bring the marketplace into common use by modeling and simulation practitioners.

2. The System Entity Structure

In this section, we review basic concepts of the SES needed in the paper.

Basic Statements for the SES

The "Made of" statement

This statement tells how a modeled component is made of, or composed from, more basic components. Alternatively, it tells how the entity representing a component is decomposed into smaller entities using the aspect relation.

```
From the <perspective> perspective, the <coupled model> is made of
<component1>, <component2>,..., and <componentN>!
```

The "Sends" (Internal Coupling) statement

A series of these statements tells how the components of a coupled model are coupled. i.e., how their output and input ports are connected together.

```
From the <perspective> perspective, <component1> sends <Message> to
<component2>!
```

or

```
From the <perspective> perspective, <component1> sends <outPort> to
<component2> as <inPort>!
```

The "Sends" (External Input Coupling) statement

A series of these statements tells how the input ports of a coupled model are coupled to input ports of its components.

The "Sends" (External Output Coupling) statement

A series of these statements tell how the output ports of a coupled model are coupled to output ports of its components.

```
From the <perspective> perspective, <component> sends <Message> to
<coupled model>!
```

The "Can be" (Specialization) statement

Specialization specifies alternative choices for a component. A component can be thought of as a place holder into which one of the alternatives can be "plugged".

```
<component> can be <alternative1>, <alternative2>, or
<alternativeN>, ... in <specialization>!
```

The Selection statement

Choice of alternatives from specializations is done in a process called pruning.[6] The basic statement in a pruning specification is the select statement:

```
select <alternative> from <specialization> for <component>!
```

Such specializations multiply to provide a combinatorial space of possible pruned entity structures. Along with other combinatorial operations described in Ref. 10,

the set of such structures constitutes the family of possible models generated by the SES. After pruning, a transformation algorithm creates the hierarchical coupled model corresponding to the pruned entity structure.

The "Merge All" statement

This statement is used to merge SESs into a target SE. The set of merged SESs is the set of existing SESs whose root entities exist as leaf entities in the target SES:

```
MergeAll from <SES>!
```

Example SES for NationalHealthCareSystems

To explain how to develop model families we start with a DEVS coupled model transformed from a simple SES and evolve to the next level where the SES generates a family of models. Consider the following SES specification:

```
From the financial perspective, NationalHealthCareSystem is made of
CareDelivery, Insurance and Financing!

From the financial perspective, Financing sends Payment to
Insurance!
```

The first statement specifies the components of NationalHealthCareSystem in a decomposition relevant to the financial perspective. The second statement specifies a coupling from the Financing component to the Insurance component. The NationalHealthCareSystem model that is generated by the transformation of the SES to a hierarchical coupled model is illustrated in Fig. 2. Here, the components are Financing, Insurance and CareDelivery and the couplings are depicted by arrows from output ports to input ports. For example, as specified, there is a coupling from the output port, outPayment of Financing to the input port, inPayment of Insurance. The figure is a snapshot of the model as produced by the MS4 Me Simulation Viewer which facilitates visually verifying components and couplings and running the model through various controls.

As mentioned above, in addition to components and couplings, an SES can include specializations that provide alternative choices for components. For example, a SES for NationalHealthCareSystem, shown in outline in Fig. 3, depicts specializations deliveryType for CareDelivery with alternatives like Hospitals, Clinics, and DiagnosticUnits. The natural language specification is:

```
CareDelivery can be Hospitals, Clinics, or DiagnosticUnits in
deliveryType!
```

Similarly, insuranceType is a specialization for Insurance with values PrivateProvided, PublicProvided, and SelfProvided.

In the pruning process, the user makes selections from specializations in an SES which eventually results in an executable hierarchical coupled DEVS model. For example, the model in Fig. 4 results from selections of EmployerFunded for Financing, PublicProvided for Insurance and Hospitals for CareDelivery.

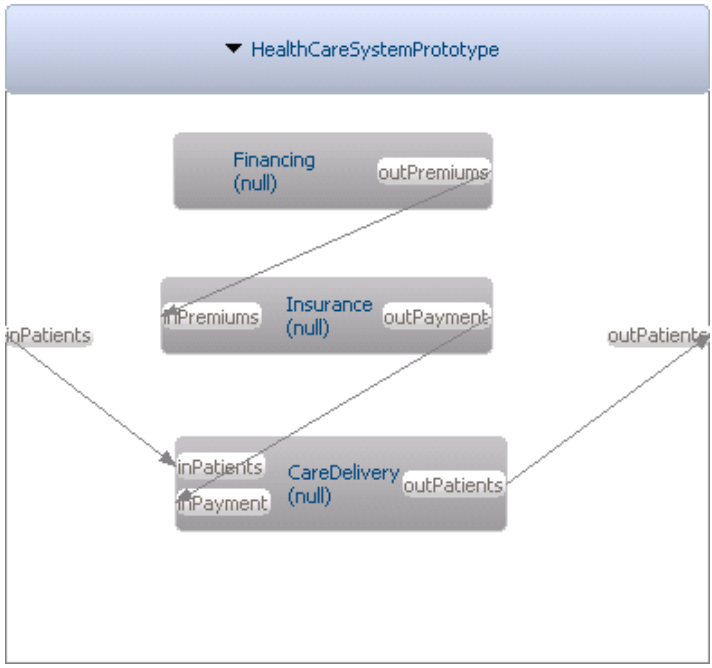


Fig. 2. NationalHealthCareSystem.

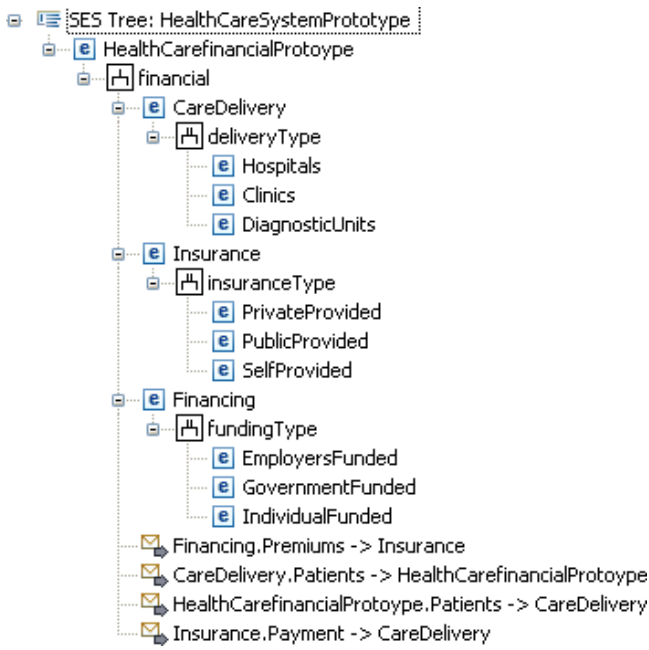


Fig. 3. Outline of SES for HealthCareSystemPrototype.

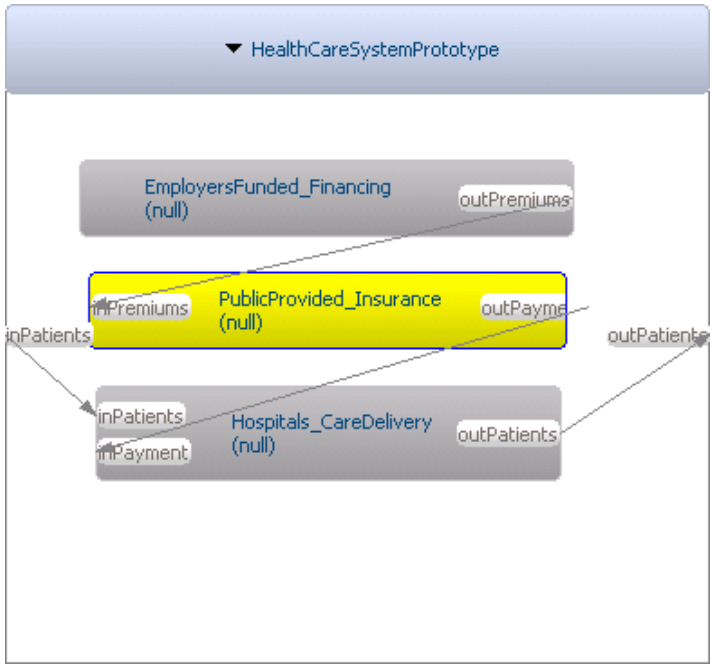


Fig. 4. NationalHealthCareSystem pruned and transformed from SES of Fig. 3.

3. Multiple Aspects

When an entity has more than one aspect, it can be decomposed in more than one way in order to represent different points of view on the same real-world system or process. The benefit of viewing the same thing from different perspectives is that this may well lead to simplifications or idealizations that allow dispensing with a lot of the complexity inherent in the real world. System abstractions derived from different perspectives often can be treated in a stand-alone or quasi-independent manner. Unfortunately, although an effective abstraction enables you to get into the right ballpark from that particular point of view, its assumptions tend to conflict with those of other abstractions when pushed beyond its limits. The advantage of including multiple aspects in a single family of SESs is that we can work with them all together as a whole when we need to.

For example, a NationalHealthCareSystem can be given another aspect based on how it might be decomposed in order to facilitate the evaluation of its place in world health organization rankings:

```
From the ranking perspective, NationalHealthCareSystem is made of
Providers,Patients, and PayerGroup!
```

Selection for aspect can be done in the pruning process similarly to the choice of specializations. For example,

```
select ranking from aspects for NationalHealthCareSystem!
```

The concept of multi-aspect provides a uniform way to associate an unlimited number of related aspects with the same entity. Each multiaspect effectively opens up a large space of simulation models with an unbounded variety of possibilities for coupling their components. For example, the entity Providers can be broken down into an arbitrary number of individual providers, each of which can be a physician, nurse, or midwife, as in:

```
From the multiProvider perspective, Providers are made of more than
one Provider!
Provider can be id in index!
Provider can be Physician, Nurse, or Midwife in careRole!
```

In pruning, the following statements will result in a set of N providers consisting of different types of individuals where N is any positive integer/

```
restructure multiaspects using index!
set multiplicity of index as [N] for Provider!
```

Since Provider can be given any SES substructure, the result of pruning would be N provider entities each of which can be pruned differently using the SES substructure.

4. Developing a Suite of Models

The process for developing, merging, pruning, and transforming an SES is illustrated in Fig. 5. Note that before pruning, the SES components of a target SES

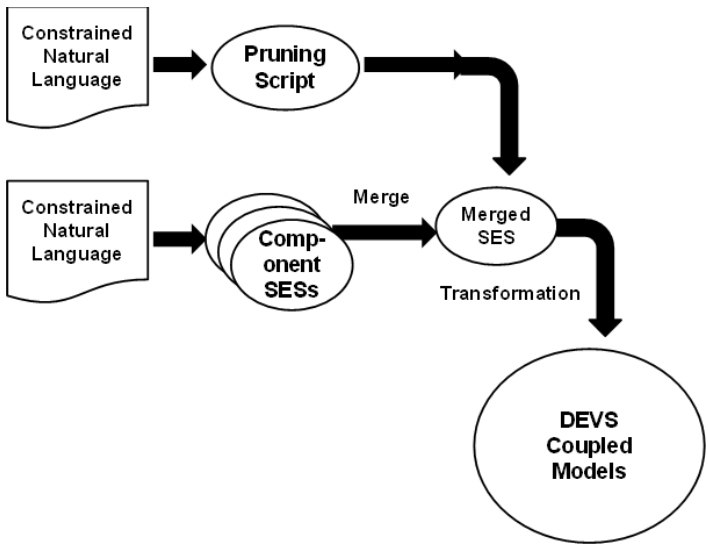


Fig. 5. Developing, merging, pruning, and transforming SESs.

are merged (recursively) to give the merged version of the target SES. The merged SES is then pruned and transformed to a hierarchical coupled model. For example, suppose that SESs exist for each of the entities, Financing, Insurance, and CareDelivery of the NationalHealthCareSystem SES. This allows Financing, Insurance, and CareDelivery to each have a family of models associated with it. For example, Financing can have different SES's for each of the types of identified funding. Then to create the merged SES for NationalHealthCareSystem, the unmerged SES of Fig. 3 is merged with the component SESs using the statement:

```
MergeAll from NationalHealthCareSystem!
```

To develop a new family of models in a suite of models, you develop the target SES for it, looking for existing models that you can use as components. Such existing models include families of models generated by SESs as well as atomic models in the repository. An existing family of models generated by an SES becomes a component SES when you terminate the top down development of the target SES at a leaf entity with the same name as this SES. This component SES will be merged into the target SES in the process illustrated in Fig. 5. For needed components that are currently not available in the suite of models, you go through the same procedure, with the additional task of recursively developing the components in the manner of the target. Of course you may decide at any point to develop a model as an atomic model rather than as one generated by an SES.

5. Marketplace of Models

To summarize, in a suite of models, each component SES represents a family of models that can be pruned and transformed to execute in a simulation. Component SESs can be merged to a new SES with the same compositional properties. This functionality leads to the concept of a marketplace of models as seen in Fig. 6.

The concept of marketplace of models extends the support for developing suites of models by using Web Services backed by Cloud Technology. MS4 Systems is developing environments to support the workflow development processes for a marketplace of models. The MS4 Store technology supplies the requisite Cloud-based model repositories. The MS4 Modeling Environment (Me) provides a range of features to enable evolution of users from neophyte to expert. For the student it provides an introduction to a simplified DEVS and Finite Deterministic DEVS. It then goes on to provide advanced support for developing full-fledged DEVS models in Java. For collaborative team developers, it provides the composition and integration facilities based on the SES described above. With the support for suites of models presented above, the MS4 Store supports families of models for combinatorial generation of architectural alternatives for exploration and optimization. The SES supports composition of models drawn from one or more model repositories. Operations on SES objects such as merging ease development by maintaining coherence of shared constructs among modifiable components. Merging enables divide-and-conquer component-based development of suites of models. Another operation,

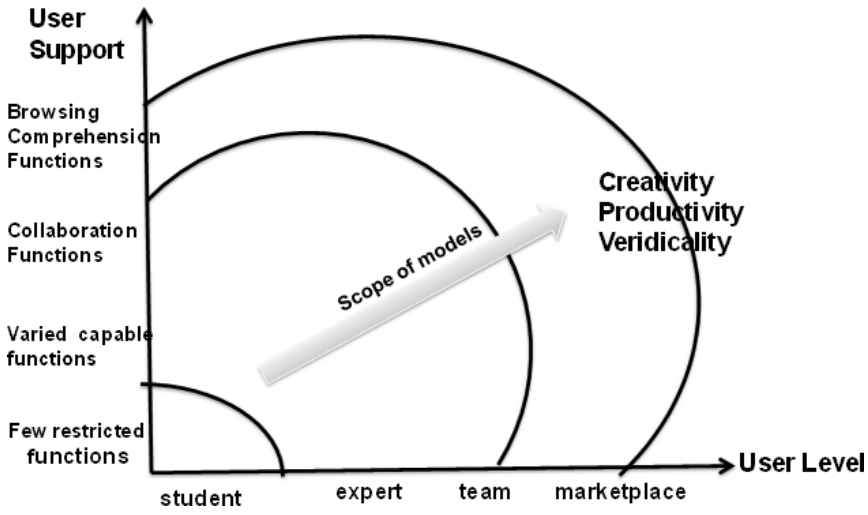


Fig. 6. Supporting a marketplace of models.

called mapping, supports tailoring and restructuring of SES components for different objectives.[5]

6. Future Extension

To go beyond team development, the Model Store must significantly extend its support for collaborative and market-oriented development of models. Such support will include browsing the SES structures of models to enable increased comprehension of model content and functionality. Such comprehension is necessary to realistically enable a developer to acquire and re-use a simulation model developed by someone else. To date, some progress has been made in making it easier for developers to provide documentation that accompanies their models as they are uploaded, However, further research is needed to develop the most efficient and effective ways of generating views of SES entities, relationships, and variables to support goal-oriented browsing. Ideally when trying to decide if an available family of models can serve your purposes, you would like to understand the SES's objectives, its role within any larger containing SES. For this, the environment should support tracing of influences forward and in reverse order to perceive the information flow and what other SES's are "friends" of the one under inspection.

7. Conclusions

This paper reviewed the SES concept and its support for developing a suite of models in the MS4 Modeling Environment. The SES can be viewed as an ontological framework for modeling and simulation.[10] In contrast to the common view of ontologies, the basic relationships in the SES relate to model structuring (aspect,

specialization, coupling, etc.) and are intended to support pruning and transformation rather than harmonization and reasoning. We described the operation of merging SESs and its key role in the methodology for such development. As an ontological framework for modeling and simulation, the SES provides the basis for other advanced operations such as mapping and tailoring, in support of reuse, composition and integration. The framework also supports browsing and comprehension of model suite structure and behavior. Further research is needed to develop the most efficient and effective ways of generating views of SES entities, relationships, and variables. After construction, the suite of models can be hosted on Model Store, the cloud-based repository of models provided by MS4 systems as a basis for collaborative and market-oriented model development.

References

1. Kim T., Lee C., Christensen E., Zeigler B., System entity structuring and model base management, *Syst. Man Cybern. IEEE Trans.* **20**(5):1013–1024, 1990.
2. Rozenblit J., Zeigler B., Representing and constructing system specifications using the system entity structure concepts, *Proc. 25th Conf. Winter simulation*, ACM, pp. 604–611, 1993.
3. Couretas J. M., Zeigler B. P., Patel U., Automatic generation of system entity structure alternatives: Application to initial manufacturing facility design, *Trans. Soc. Comput. Simul.* **16**:173–185, 1999.
4. Hagendorf O., Pawletta T., A framework for simulation based structure and parameter optimization of discrete event systems, in *Discrete-Event Modeling and Simulation: Theory and Applications*, Gabriel A. W., Pieter J. M. (Eds.), CRC Press, 2010.
5. Tuncer I. Ö., Zeigler B. P., System theoretic foundations of modeling and simulation: A historic perspective and the legacy of A. Wayne Wymore, *Simulation* **88**(9):1033–1046, 2012.
6. Zeigler B. P., Seo C., Coop R., Kim D., Creating suites of models with system entity structure: Global warming example, *Symp. Theory of Modeling & Simulation – DEVS (TMS/DEVS) Spring Sim 2013*, San Diego, CA, USA, 2013.
7. Seo C., Zeigler B. P., Coop R., Kim D., DEVS modeling and simulation methodology with MS4Me software, *Symp. Theory of Modeling & Simulation – DEVS (TMS/DEVS) Spring Sim 2013*, San Diego, CA, USA, 2013.
8. Zeigler B. P., Hammonds P., *Modeling & Simulation-Based Data Engineering*: *Introducing Pragmatics into Ontologies for Net-Centric Information Exchange*, Academic Press, Boston, p. 448, 2007.
9. Zeigler B. P., Sarjoughian H. S., *Guide To Modeling and Simulation of Systems Series: Simulation Foundations, Methods and Applications*, Springer Pub. Co., p. 330, 2013.
10. Verbraeck, Component-based distributed simulations. The way forward? *Proc. 18th Workshop on Parallel and Distributed Simulation (PADS'04)*, pp. 141–148, 2004.
11. Rohl M., Uhrmacher A. M., Composing simulations from xml-specified model components, *Proc. Winter Simulation Conf.* 06, ACM, pp. 1083–1090, 2006.

Biography

Bernard P. Zeigler is Chief Scientist with RTSync Corp, Research Professor with the C4I Center of George Mason University, and Emeritus Professor of Electrical

and Computer Engineering at the University of Arizona. He is internationally known for his 1976 foundational text *Theory of Modeling and Simulation*, revised for a second edition (Academic Press, 2000). He has published numerous books and research publications on the Discrete Event System Specification (DEVS) formalism. In 1995, he was named Fellow of the IEEE in recognition of his contributions to the theory of discrete event simulation.

Chungman Seo is a senior research engineer at RTSync Company and a member of Arizona Center for Integrative Modeling & Simulation (ACIMS). He received his Ph.D. in Electrical and Computer Engineering from The University of Arizona in 2009. His research includes DEVS-based web service integration, DEVS/SOA-based distributed DEVS simulation, and DEVS simulator interoperability.

Doohwan Kim is the founder and president of RTSync and MS4 Systems. He is the first to introduce the commercial innovation of the DEVS modeling and simulation software toolsets and Cloud-based model store environment. Previously, Dr. Kim was working on M&S projects at IBM T. J. Watson Research Center, NY and held a research professorship at the University of Arizona. He received his M.S. and Ph.D. from the University of Arizona.