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.

A SIMULATION OPTIMIZATION FRAMEWORK FOR DISCRETE EVENT LOGISTICS SYSTEMS (DELS)

Timothy Sprock
Leon F. McGinnis

Department of Industrial and Systems Engineering
Georgia Institute of Technology
Atlanta, GA 30332 USA

ABSTRACT

For large-scale, complex systems, both simulation and optimization methods are needed to support system design and operational decision making. Integrating the two methodologies, however, presents a number of conceptual and technical problems. This paper argues that the required integration can be successfully achieved, within a specific domain, by using a formal domain specific language for specifying instance problems and for structuring the analysis models and their interfaces. The domain must include a large enough class of problems to justify the resulting specialization of analysis models.

1 INTRODUCTION

Increasing scale, complexity, and connectedness have accentuated a need to improve decision-making support in Discrete Event Logistics Systems (DELS), a class of dynamic systems that create value by transforming discrete flows through operations performed by a network of interconnected subsystems. The DELS domain includes systems such as supply chains, manufacturing systems, transportation networks, warehouses, health care delivery systems, etc. Decision-making support for these systems can be integrated into many aspects of DELS from system design decisions, such as network configuration and resource selection, to online, real-time operational control decisions, such as dispatching and routing. To simultaneously address the complexity and scale of these systems, simulation and optimization offer complementary views of the system model; simulation is more adept at capturing and evaluating the dynamic behavior of the system, while optimization is more effective at tackling the scale and searching the design space of the system. Consequently, simulation optimization methods, which in their most general form aim to efficiently use simulation integrated with search algorithms, seem like a promising approach for providing decision support for this domain. In fact, simulation optimization methods have been applied to solve a number of problems in this domain; for a representative sampling see, e.g., supply chains (Truong and Azadivar 2005), manufacturing (Kapuscinski and Tayur 1999), transportation (Cheng and Duran 2004), warehouses (Rosenblatt, Roll, and Vered Zyser 1993), healthcare (Ahmed and Alkhamis 2009), and emergency logistics (Ng, Park, and Waller 2010).

Despite the evidence that simulation optimization is applicable, several general requirements, or challenges, must be more fully addressed by the research community to make simulation optimization a routine analysis. Despite more than fifteen years since the identification of this fundamental issue, the majority of the methodological research on simulation optimization continues to focus on a single aspect of simulation optimization (either the simulation or the optimization) without considering the subject as a whole (Bowden and Hall 1998). The academic literature treats the optimization modeling and simulation modeling as two different and distinct activities and often there is not an ‘equal partnership’ between the two tools (Fu 2002). In addition to these methodological challenges, there are several practical challenges to overcoming the gap between academic research and practical implementations (Fu 2002), including difficulty interfacing simulation and optimization (Azadivar 1999), generating and evaluating large scale simulations, interfacing with data sources and execution systems.

In order to work on large and complex applications, individual researchers need to be able to focus on their own particular research (either simulation or optimization) and trust that interoperability is available with effective implementations of other analysis components. Creating this interoperability between the various methodologies and their associated tools, from global optimization methods to high-performance simulation methods remains a largely unmet challenge. To address this challenge, Industrial Strength Compass introduces a simulation optimization platform that integrates global, local, and ranking and selection methods through an interface that exchanges solutions and feedback with simulations (Xu, Nelson, and Hong 2010). Several authors standardize the interface between the simulation and optimization tools through the exchange of a generic problem definition object (Duvivier et al. 2003, Almeder et al. 2008, Hamm et al. 2011, Pasupathy and Henderson 2011). Commercially, the success of OptQuest$^{\text{TM}}$can be partially be attributed to its integration with a multitude of popular simulation tools. However, customizing OptQuest for each simulation platform is an expensive process that doesnt scale well in a many-solver environment (Fu et al. 2014). Despite these advances, the challenge remains to provide a universal solution that supports plug-and-support functionality for the broader research community.

In the research presented here, we propose utilizing formal domain modeling methods as part of a generic solution to this interoperability problem. A formal model of the domain is an important concept because it allows research on system design and analysis to be implemented independently of a specific system instance model but with some guarantee of interoperability between components. Simulation and optimization tools which are not based on any canonical description, of course, can be created and used to model and analyze systems, but differences among the tools in terms of semantics, syntax, and organization will create friction and obstacles to sharing, reuse, and widespread adoption. What we aim to show is that it is possible to specify a formal model for a sufficiently large problem domain to justify the investment in the domain model and the associated modeling infrastructure.

In this paper, we propose a simulation optimization framework for the DELS domain rooted in a formal model of the domain. This formal domain modelling methodology uses a domain specific language (DSL) to construct system and analysis models and the associated tools. Section 2 introduces the formal domain modelling approach for DELS. In particular, section 2.1 briefly introduces some of the features that need be incorporated into a formal model of the DELS domain, section 2.2 discusses the benefits of specialized analysis methods, section 2.3 introduces the formal domain modeling method, and finally section 2.4 highlights the desired characteristics of the formal model. Then Section 3 presents a use case that drives the analysis modeling methodology from a formal model of the system captured in SysML. Finally, Sections 4 and 5 conclude with discussion on applications, limitations, and future work.

2 FORMAL DOMAIN MODELING FOR DELS

2.1 Domain-Specific Features of DELS

One of the primary reasons for constructing domain specific analysis methodologies is to exploit domain characteristics that allow the methods to be much faster, cheaper or better than their generic counterparts. For DELS, the optimization process can be adapted to handle the multitude of sources of uncertainty and complex system dynamics inherent to these systems, multiple objectives and categorical variables which are prominent features of this class of problems, and multiple time scales and planning horizons which lead to many interrelated decision problems.

In addition to opportunities to improve the simulation execution engines, there is also the need to reduce the cost and complexity of simulation development including improving conceptual modeling methods as well as reducing the development barrier through model and component reuse (Robinson 2005). Related to conceptual modeling there are questions about which type of simulation is appropriate, whether it be petri nets, agent based simulation, discrete event simulation, etc. There are also issues related to what information is available as well as the fidelity and level of aggregation of the simulation, including when and how to make approximations about the system. Several examples are presented in the next section that demonstrate methods to address these domain specific challenges and allude to modeling requirements for domain specific modeling language for DELS.

2.2 Why Specialize?

Obviously, there is a trade-off between constructing methods that exploit specific characteristics of a particular problem instance and developing more general methodologies capable of handling a diverse collection of problem classes. Specialized algorithms based on an explicit model of a particular domain not only support the development of integrated or hybrid methods for that domain, but they can also guarantee consistency in comparing the performance of competing algorithms (Glover et al. 1995, Xu et al. 2010). However, the specialization requires an investment in research and development.

Many optimization algorithms can be tailored to exploit domain specific characteristics. In genetic algorithms, the domain specific structure can be exploited through specialized encoding schemes, initialization, and local search operators. The structure of each chromosome is segmented to independently represent components of the problem that reflect the network structure, the resource selection problem, and the control policy selection and configuration; e.g. (Dengiz et al. 1997, Zhou et al. 2002, Syarif et al. 2002, Costa et al. 2010). In Tabu methods, domain specific search neighborhoods can be integrated into the local search procedure, e.g. swaps of orders within the schedule (Yang et al. 2004), separation of the routing and scheduling components of the job shop problem (Brandimarte 1993), and setting Kanban levels in a production system. Finally, knowledge-based optimization methods incorporate learning modules that integrate domain specific information to guide the optimization search process, such as selecting between priority rules and lot sizing (Huyet and Paris 2004).

From these examples, it becomes clear that many industrial, service, and other complex systems can be characterized by their structural designs, operational behavior, and control policies. For these applications, these characteristics can be exploited to provide narrowly-scoped and well-structured search neighborhoods that can correspond to modifying the network structure, resource investment, or operational policies independently (Azadivar 1999, Azadivar and Tompkins 1999, Ding et al. 2009, Costa et al. 2010). Furthermore, Costa et al. (2010) present evidence of a formal model that defines the network structure separately from the set of control decisions and policies, as well as the definition of domain specific performance indicators. Often modifying the network structure or categorical variables associated with control policies requires making structural modifications to the simulation throughout the optimization process thus requiring the simulation to be regenerated at each step (Azadivar and Tompkins 1999).

To achieve interoperability and consistency between domain specific algorithms, the algorithms or at least the system that they are optimizing must be derived from an agreed-upon and explicit system definition that not only supports methods to generate the required set of analysis models, but also is capable of translating optimization outputs into instructions to modify or generate the desired simulation for execution.

2.3 Formal Domain Modeling

In the research presented here, formal domain modeling methodologies are used to improve the interoperability between research methods. A formal model of the domain consists of an explicit language for describing components of a system for a particular domain, a domain specific language (DSL), and rules for assembling those components into meaningful and accurate models of the system. While the DSL is intended to capture the syntax of a particular domain, the reference architecture for a specific domain captures implicit knowledge that is commonly useful to stakeholders in that domain. For simulation optimization methodologies, the formal domain model enables the specification of formal models of both the system optimization and system simulation with a common ontological base, thereby providing a critical requirement for analysis model interoperability.

Several researchers have recognized the importance of creating object-oriented reference architecture types of conceptual models for the supply chain domain; for an overview of many of these frameworks, see (Grubic and Fan 2009). Many of these frameworks exclusively target the simulation analysis domain: (Jain et al. 2001, Chatfield et al. 2006, Rossetti et al. 2008). Object-oriented models provide a flexible and reusable framework for capturing the domain; (Narayanan et al. 1998, Biswas and Narahari 2004, Kim and Rogers 2005). These frameworks typically lack a formal language implementation, which limits the full realization of the benefits of developing the reference architecture, such as extensibility and reusability. Azadivar and Tompkins (1999) present a methodology that captures the system description as a System Model Object, which separates the network definition from the control policies. Their methodology suggests a generalizable strategy where the optimization problem and corresponding simulation model are both generated automatically through a transformation from the system model.

There is an outstanding need for a domain model for the class of DELS that consists of a DSL for capturing a broad family of systems. The challenge is to develop the DSL at a sufficient level of abstraction that it is broad enough to justify the investment in domain specific tools and methods, while also being specific enough to drive productivity enhancements from these modelling and analysis tools and methods.

2.4 A Formal Domain Model for Discrete Event Logistics Systems (DELS)

Our formal model of the DELS domain is constructed as a multi-layer architecture which captures the underlying network structure, the behavioral elements, and the control policies of the system. The separation and encapsulation of the layers is important to provide multiple degrees of abstraction of the system model to formulate specific types of analysis models. These layers of abstraction formalize some of the aspects that are noted in the specialization section and often correspond with the sequential design strategies or partitioning of chromosomes in genetic algorithms.

The network, or graph, is a common abstraction across operations research as an abstraction of both system models and analysis models. The token flow network is a multi-layer architecture that starts by formalizing basic network semantics of node and edge and then extends that architecture to support flow semantics, such as those expected to support the description of a multi-commodity network flow abstraction (MCFN) (Thiers 2014). Finally since DELS commonly transform these discrete flows, the flow network semantics are augmented with behavioral semantics, such as state machines, to support the utilization of resources to convert a set of inputs into a set of outputs (Sprock and McGinnis 2014). In addition the underlying network definition can be extended with the additional semantics needed to specify DELS models, including a common description of the system, addressing its structure and behavior, its owned and shared resources, and its interaction with other systems. Specifically, extensions are needed to specify what is being transformed (product), how is the product transformed (process), what is executing the transformation (resource), and where do the resource live and how are they configured (facility). The product and process descriptions are most commonly expressed as a bill of material and a process plan (bill of process), respectively. The semantics of this layer draw inspiration from manufacturing reference architectures, such as MASON and CMSD (Lemaignan et al. 2006, Lee et al. 2011). In addition to capturing the structure and behavior of the system plant, the reference architecture must also include a control model that is layered on top to provide interaction and influence over the activities executed throughout the DELS.

3 TRANSLATING MODELING METHODOLOGY INTO ANALYSIS METHODOLOGY: A USE CASE

In this section, we demonstrate a proof of concept usage of a DSL for DELS to construct domain specific analysis methods and tools. This demonstration is not intended to be a contribution or competitor to existing simulation optimization implementations, but to highlight some features of a formal domain specific analysis methodology, such as developing robust interfaces to analysis tools that are based on the system model and formulating analyses to exploit specific characteristics of the system model, such as its structure, behavior, or control.

3.1 Distribution Supply Chain System Model

A distribution supply chain design problem is used to demonstrate the proposed framework (Figure 1). Given a set of customers that seek to ship commodities to one another, the systems architect must locate a set of depots (from a discrete set), size the fleet of trucks for each depot, and select a control policy to dispatch the fleet of trucks. The distribution network is composed of Depots and Customers that are connected via Transportation Channels, where the Depots can dispatch Trucks, or transportation resources, to Customers to drop off or pick-up Shipments. There are fixed costs to opening a Depot and purchasing Trucks for each Depot, as well as variable costs for traversing Transportation Channels. Finally, a commodity is defined for each customer-customer relationship and shipments between customers are promised within a particular lead-time.

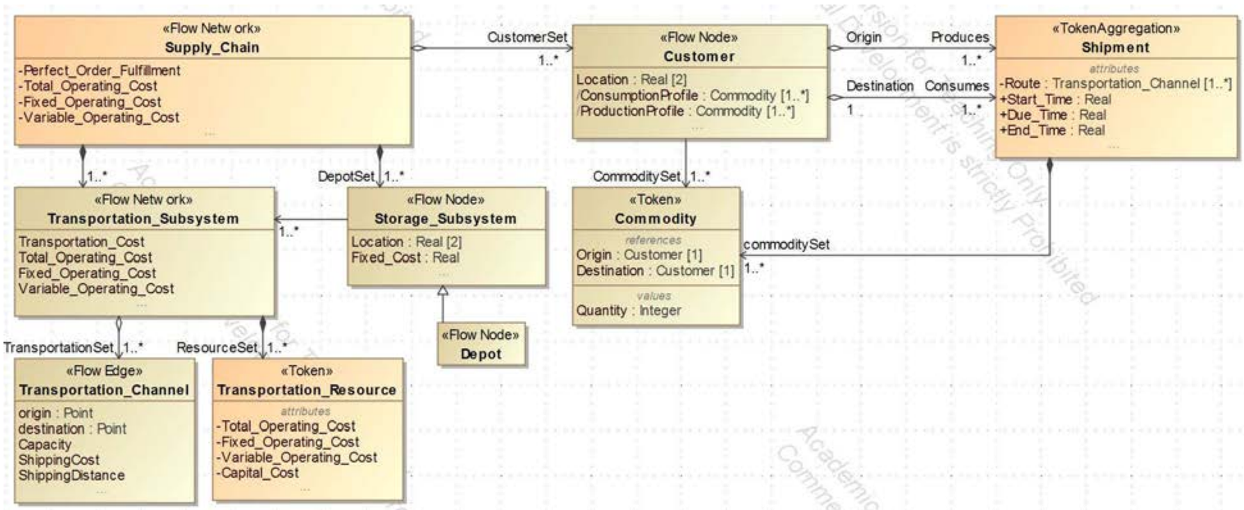

Figure 1: Distribution Supply Chain System Model.

In addition to specifying the structure of the system, its behavior must also be captured in the system model (in a set of behavioral diagrams separate from Figure 1). Once a truck is dispatched from a Depot to service a Customer, it will both drop-off all shipments destined for that Customer and pick-up Shipments that need to be routed through the Depot. The same is true for transportation from Depot to Depot, but either Depot can allocate a truck to perform the Service. Each Depot is approximated as a cross-dock with no processing requirements and simply routes each shipment to its outbound transportation queue to await a truck. Finally, a control policy must specify how trucks are allocated to transportation channel, i.e. which customer to service next. The routing of shipments through the system is done passively according to the Route, a sequence of Transportation Channels, which is constructed during the optimization routines.

A useful way to think about the model presented in Figure 1 is that it presents a schema to a database that holds all of the system data. If simulation optimization methods and tools, and analysis methods in general, are constructed to operate and interact with the data conforming to the schema, then any system description that conforms to the schema will be usable by those methods and tools. One challenge of this approach is to design tools that specifically pass data conforming to the schema defined by the system model rather than context-free arrays. The next three sections will present current efforts to design tools that are capable of exploiting specific characteristics of the system model, specifically the underlying network abstraction, resource investment problem, and policy selection and configuration.

3.2 Global Search Based on a MCFN Approximation

The first stage of this analysis maps the key elements of the system model to a flow network model in order to formulate an analysis based on a multi-commodity flow network (MCFN). This abstraction process allows analysis tools to be constructed using the DSL presented in Figure 2, and then applied to a system model such as the *FlowNode* and *FlowEdge* stereotypes in Figure 1. Since the system model can be abstracted to a flow network definition, it can utilize any set of analysis tools constructed for flow network analysis such as multi-commodity network flow models formulated for solution by CPLEX.

While the Industrial Strength Compass tool-set discussed above uses a niching genetic algorithm for the global search stage, Osman suggests heuristics methods, including a branch and bound method from a MIP solver, to generate feasible, diverse, and good solutions to jump-start the local optimization process (Osman 1995). This strategy utilizes the strength of modern day commercial solvers, such as CPLEX, at any reasonable level of approximation of the system to produce a collection of candidate backbone solutions to the system. This approximation is useful for quickly discarding inferior solutions, such as in supplier selection or facility location problems.

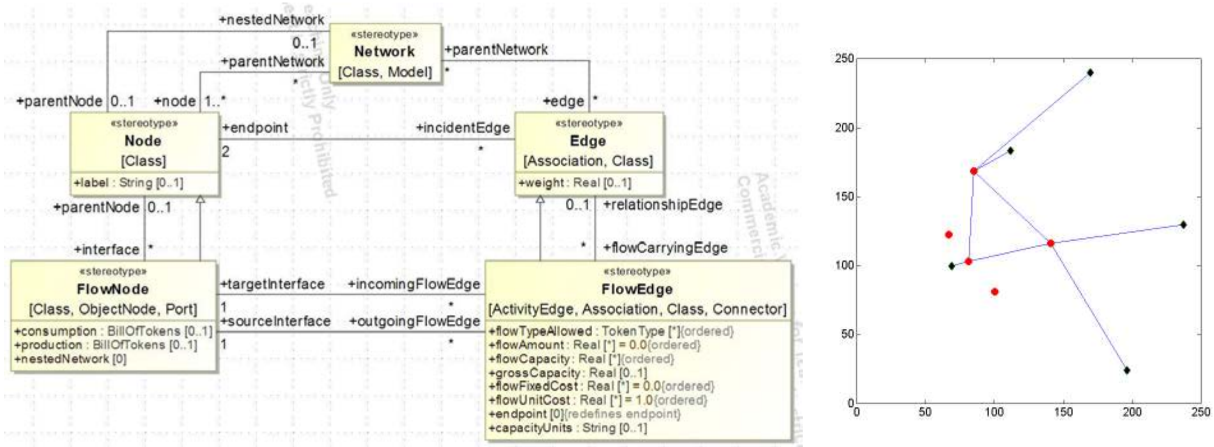

Figure 2: Flow network abstraction (Left) and results of multi-commodity flow network optimization (Right) with depots highlighted in red circles.

In this use case, the output of this stage is a selected subset of depots, the assignment of customers to depots, and routes for the commodities to flow through the network. While the optimal solution to the MCFN is presented in Figure 2, the MIP can also return the last k solutions to populate an initial pool of candidate solutions. The analysis shown here utilizes a simpler leave one out heuristic, which re-solves the problem iteratively without one of the depots selected in the optimal solution.

3.3 Resource Investment via Genetic Algorithm

Whereas the MCFN analysis is deterministic approximation of the system model, the simulation tool is more adept of modeling the dynamic behavior, complex interactions, and variability inherent to the system. Therefore, it is reasonable incorporate these effects into a methodology that utilizes simulation as its evaluation tool. However, first, the results from the MCFN approximation need to be translated back into a set of complete system models. These system models are then used to generate discrete event simulation in SimEvents®(Figure 3) using an object oriented, network-based transformation engine (Sprock and McGinnis 2014). Then each simulation is embedded within a multi-objective genetic algorithm from the MATLAB global optimization toolbox. The multi-objective genetic algorithm in MATLAB uses a controlled elitist GA variant of NSGA-II to converge to an optimal Pareto frontier (Deb et al. 2002).

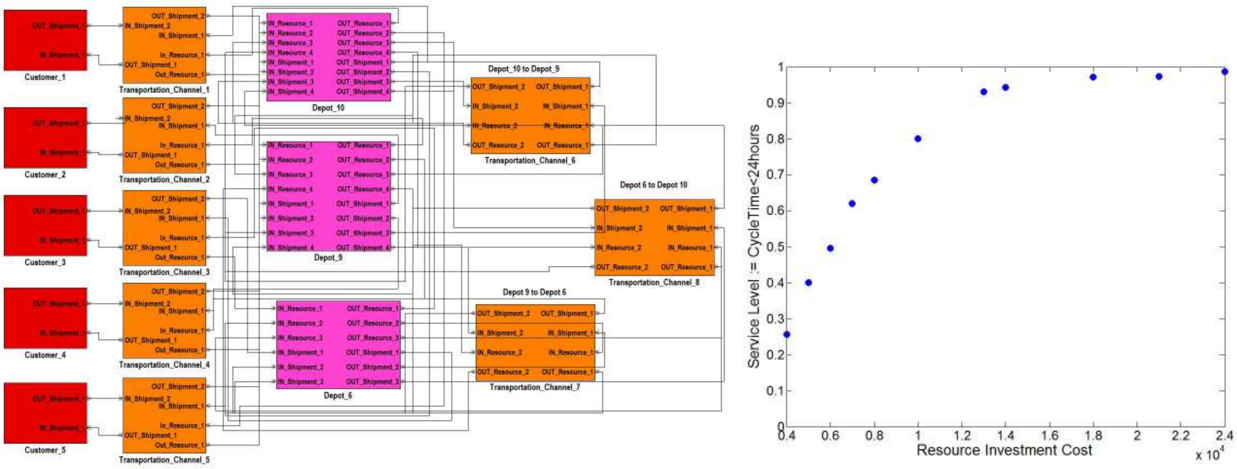

Figure 3: Generated Discrete Event Simulation in SimEvents (Left) and Output of Multi-Objective Genetic Algorithm (Right).

There are two extremes for defining the interface between the simulation and optimization. The first extreme passes a solution to a pre-defined simulation for evaluation (Pasupathy and Henderson 2011). This requires a hard-coded, and often implicit, mapping between the solution vector and the associated variable in the simulation. At the other extreme, the interface between the simulation and optimization is defined by a system model object and an experiment definition and the solution is returned in the system model object (Hamm et al. 2011). This would allow the optimization routine to utilize any attributes of the system defined or captured in the simulation process, not necessarily only the evaluated value itself; e.g. examining resource utilization in a system that seeks to balance the trade-off between make-span and cost. Since this is computationally expensive, in our presented use case the evaluation function embedded in the GA defines an interface that accepts a subset of the data that is sufficient to construct and evaluate the candidate solution.

In this use case, the GA is tasked to evaluate the trade-off between Resource Investment, how many trucks to purchase for each depot, and the Service Level, % of shipments completed in under 24 hours. The result is a Pareto set of solutions that can be passed to the next optimization stage (Figure 3). The model assumes a basic dispatch control policy that allocates trucks to pick-up and drop-off routes in a round-robin manner, an assumption that will be isolated and evaluated in the third stage.

3.4 Control Policy Selection via Enumeration

Throughout the course of this analysis, the system has been modelled as a passive flow network. One of the outstanding challenges in the design process remains how to design and implement the control mechanisms. The last stage of the simulation optimization evaluates a small collection of policies that control the dispatch of trucks to transport shipments. The most basic scenario uses a *round-robin policy*, which is easy to implement due to its minimal information requirements, but risks under-utilizing the capacity of each truck. The second scenario uses a *longest queue policy*, which requires the controller to gather and evaluate queue lengths from each of its customers but still risks under-utilizing the capacity of each truck. Finally, the third scenario sets *minimum queue length* before dispatching a truck to a particular customer. This improves the utilization of each truck and reduces overall distance travelled, but may increase the cycle time for shipments. This stage enumerates the complete set of resource investment solutions from the last stage for each of the control policies.

Simulation languages often have a small set of modeling constructs that implement control rules; e.g. queues that prioritize the next job to be processed or routing blocks that can decide which output port to send a job or resource. However, a canonical model of control for the DELS domain, especially one that can be implemented in DES tools, remains elusive. That is, there does not exist a universally accepted model of control decisions that can be made by a DELS or a method to implement those decisions in tools without resorting to ad-hoc code. These challenges make defining and searching the control policy space difficult, which often to leads to simplifications such as exploring a pre-defined set of control policies, such as FIFO in queues or round-robin routing. Additional work is needed in the future to standardize the specification and implementation of control policies in DELS.

3.5 Results

The output of this multi-stage simulation optimization process is a Pareto set of solutions, which can be further refined by state of the art ranking and selection methods. However, in its current state the output is a trade-off curve between resource investment (depots and trucks), total distance travelled to service customers, and the service level (cycle time satisfaction). The set of solutions is projected into three two-dimensional plots (Figure 4).

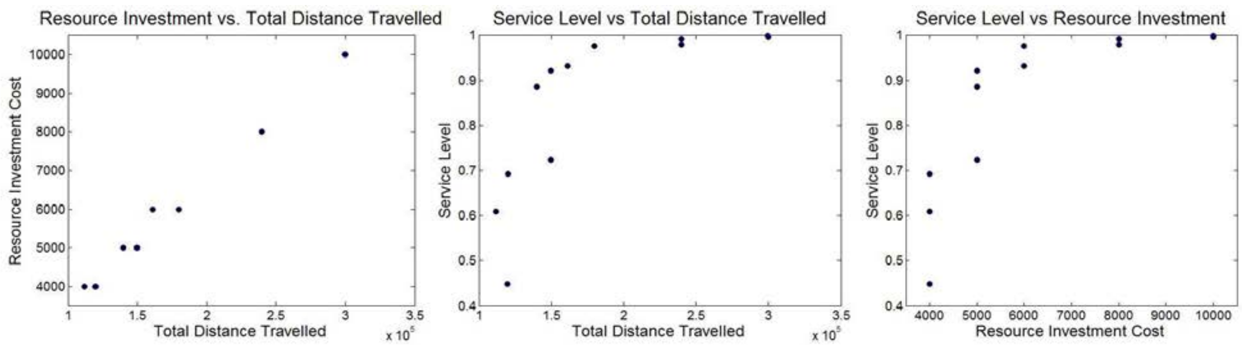

Figure 4: Candidate solutions from multi-objective simulation optimization.

3.6 Extensions to Support Additional Use Cases

Justifying the development of a formal domain methodology for simulation optimization requires demonstrating that the domain model and associated analysis methodology can be extended to support the analysis of a broad range of related systems. In fact the token flow network and product, process, resource, and facility definitions (see section 2.4 for an overview) provide sufficient detail to specify any DELS. While an exhaustive demonstration on this claim is beyond the scope of this paper, a manufacturing design case is presented to provide some detail on the extensibility of this methodology.

In the design of a manufacturing flow line or a job shop (Figure 5), there may exist several products with their own unique process plans, required processing capabilities, and resources in the form of fixtures, machines, subcomponents, etc. In the first stage, the structure of the manufacturing line or shop is constructed by extracting the underlying process and flow networks from the problem definition and allocating processing tasks to flow nodes. This allocation can be based on the grouping of capability requirements or through an assembly-line balancing method. Next, the design methodology seeks to trade-off the resource investment costs against the desired service level, cycle time, throughput, etc. In this level of the hierarchy, a MCGA evaluates resource levels for operators, required fixtures, and parallel machines at a workstation. Finally, the last stage in the design hierarchy searches for optimal inventory policies for subcomponents at each workstation and enumerates dispatch policies for releasing work into the system.

Therefore, by mapping the problem statement to the reference architecture, the design space can be structured to optimize the structure, behavior, and control of the system using domain specific methods.

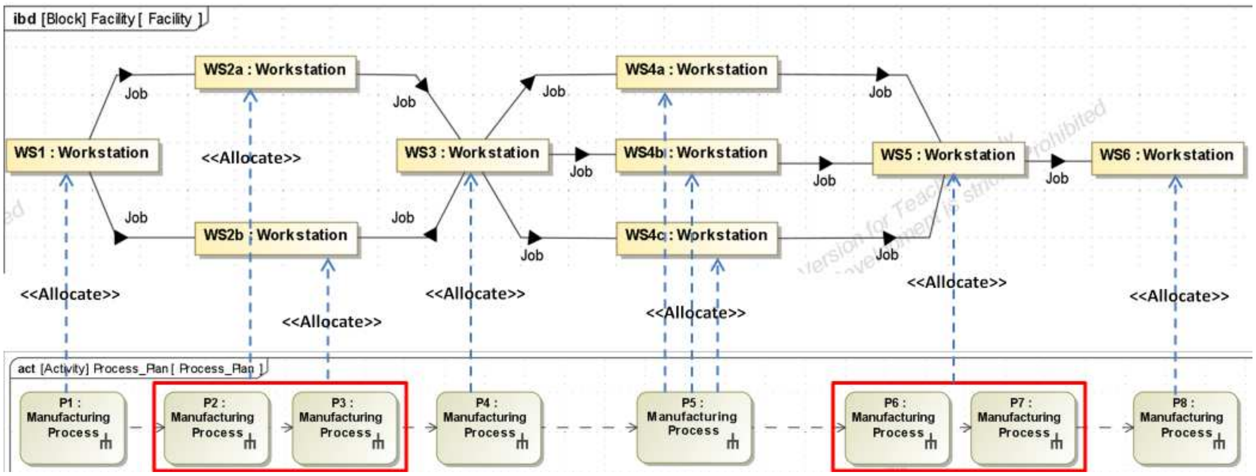

Figure 5: Manufacturing Configuration Model Captured in SysML.

The next section contains discussion on the extensibility and limitations of this methodology, specifically systems that require a different hierarchy.

4 DISCUSSION

While this methodology is used extensively in other engineering disciplines where model-based design is more routine (Sobieszczanski-Sobieski and Haftka 1997), there remain some open questions about its applicability and effectiveness in the DELS domain. Foremost, these questions stem from limited usage of model-based methods in the DELS domain and consequently incomplete reference models for the domain. While the refinement of reference models is a work in progress, there remains a need for a canonical model of control that is uniformly applicable across each of the subdomains listed in the introduction.

Additionally, hierarchical design is a relatively unexplored method in the DELS literature beyond bi-level optimization. Therefore, there are several related open questions that may impact the overall applicability of the methodology, including: Is there only one, or a best design hierarchy? How are surrogates or approximations selected, and can they be reused across sub-domains or are they sub-domain specific? How can models be generated with varying degrees of fidelity from the same base model, such as in (Jin 2011)? Finally, there are implementation issues, such as management of component variants and composability among analysis components, which may prove a challenge to verifying and validating the resulting analysis.

5 CONCLUSIONS AND FUTURE WORK

To meet the analysis requirements of next-generation discrete event logistics systems, simulation optimization methods must be able to handle the scale, complexity, and uncertainty inherent to these systems. Integration and interoperability between the individual methods within the simulation optimization methodology is a key enabler to a broader effort to meet those requirements. In this paper, we have proposed a formal domain modeling methodology to create domain specific methods and robust interfaces between those methods. For the DELS domain, this methodology suggests the use of a domain-specific language that supports the specification of the structural, behavioral, and control aspects of each system. This approach is demonstrated through a distribution supply chain use case that integrates CPLEX, a multi-objective genetic algorithm, and a discrete event simulation tool SimEvents.

Ongoing research is focused on refining the domain specific language and associated reference model of the DELS domain. This refinement process is focused on creating a language that is broadly applicable to the whole domain and establishes a basis for creating conforming simulation optimization tools. Future work will focus on tailoring this domain specific simulation optimization methodology to provide on-line real-time control for smart operational control in DELS as well as to support design methodologies for the domain.

REFERENCES

Ahmed, M. A., and T. M. Alkhamis. 2009. "Simulation Optimization for an Emergency Department Healthcare Unit in Kuwait". *European Journal of Operational Research* 198 (3): 936–942.

Almeder, C., M. Preusser, and R. F. Hartl. 2008. "Simulation and Aptimization of Supply Chains: Alternative or Complementary Approaches?". *OR Spectrum* 31 (1): 95–119.

Azadivar, F. 1999. "Simulation Optimization Methodologies". In *Proceedings of the 1999 Winter Simulation Conference*, edited by P. A. Farrington, H. B. Nembhard, D. T. Sturrock, and G. W. Evans, 93–100. New York, NY, USA: ACM.

Azadivar, F., and G. Tompkins. 1999. "Simulation Optimization with Qualitative Variables and Structural Model Changes: A Genetic Algorithm Approach". *European Journal of Operational Research* 113 (1): 169–182.

Biswas, S., and Y. Narahari. 2004. "Object Oriented Modeling and Decision Support for Supply Chains". *European Journal of Operational Research* 153 (3): 704–726.

Bowden, R. O., and J. D. Hall. 1998. "Simulation Optimization Research and Development". In *Proceedings of the 1998 Winter Simulation Conference*, edited by D. J. Medeiros, E. F. Watson, J. S. Carson, and M. S. Manivannan, 1693–1698. Los Alamitos, CA, USA: IEEE Computer Society Press.

Brandimarte, P. 1993. "Routing and Scheduling in a Flexible Job Shop by Tabu Search". *Annals of Operations research* 41 (3): 157–183.

Chatfield, D. C., T. P. Harrison, and J. C. Hayya. 2006. "SISCO: An Object-oriented Supply Chain Simulation System". *Decision Support Systems* 42 (1): 422–434.

Cheng, L., and M. A. Duran. 2004. "Logistics for World-wide Crude Oil Transportation Using Discrete Event Simulation and Optimal Control". *Computers & chemical engineering* 28 (6): 897–911.

Costa, A., G. Celano, S. Fichera, and E. Trovato. 2010. "A New Efficient Encoding/Decoding Procedure for the Design of a Supply Chain Network with Genetic Algorithms". *Computers & Industrial Engineering* 59 (4): 986–999.

Deb, K., A. Pratap, S. Agarwal, and T. Meyarivan. 2002. "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II". *IEEE Transactions on Evolutionary Computation* 6 (2): 182–197.

Dengiz, B., F. Altiparmak, and A. E. Smith. 1997. "Local Search Genetic Algorithm for Optimal Design of Reliable Networks". *IEEE Transactions on Evolutionary Computation* 1 (3): 179–188.

Ding, H., L. Benyoucef, and X. Xie. 2009. "Stochastic Multi-objective Production-Distribution Network Design using Simulation-based Optimization". *International Journal of Production Research* 47 (2): 479–505.

Duvivier, D., V. Dhaevers, V. Bachelet, and A. Artiba. 2003. "Integrating Simulation and Optimization of Manufacturing Systems". *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on* 33 (2): 186–192.

Fu, M. C. 2002. "Optimization for Simulation: Theory vs. Practice". *INFORMS Journal on Computing* 14 (3): 192–215.

Fu, M. C., G. Bayraksan, S. G. Henderson, B. L. Nelson, W. B. Powell, I. O. Ryzhov, and B. Thengvall. 2014. "Simulation Optimization: A Panel on the State of the Art in Research and Practice". In *Proceedings of the 2014 Winter Simulation Conference*, edited by A. Tolk, S. D. Diallo, I. O. Ryzhov, L. Yilmaz, S. Buckley, and J. A. Miller, 3696–3706. Piscataway, NJ, USA: IEEE Press.

Glover, F., J. P. Kelly, and M. Laguna. 1995. "Genetic Algorithms and Tabu Search: Hybrids for Optimization". *Computers & Operations Research* 22 (1): 111–134.

Grubic, T., and I. S. Fan. 2009. "Integrating Process and Ontology for Supply Chain Modelling". In *Interoperability for Enterprise Software and Applications China, 2009. IESA'09. International Conference on*, 228–235.

Hamm, M., K. Szczesny, V. Nguyen, and M. Knig. 2011. "Optimization of Construction Schedules with Discrete-event Simulation Using an Optimization Framework". In *Proc. of the 2011 ASCE International Workshop on Computing in Civil Engineering, Miami, FL, USA*.

Huyet, A., and J. Paris. 2004. "Synergy Between Evolutionary Optimization and Induction Graphs Learning for Simulated Manufacturing Systems". *International Journal of Production Research* 42 (20): 4295–4313.

Jain, S., R. W. Workman, L. M. Collins, E. C. Ervin, and A. P. Lathrop. 2001. "Supply Chain Applications II: Development of a High-level Supply Chain Simulation Model". In *Proceedings of the 2001 Winter Simulation Conference*, edited by B. A. Peters, J. S. Smith, D. J. Medeiros, and M. W. Rohrer, 1129–1137. Washington, DC, USA: IEEE Computer Society.

Jin, Y. 2011. "Surrogate-assisted Evolutionary Computation: Recent Advances and Future Challenges". *Swarm and Evolutionary Computation* 1 (2): 61–70.

Kapuscinski, R., and S. Tayur. 1999. "Optimal Policies and Simulation-based Optimization for Capacitated Production Inventory Systems". In *Quantitative Models for Supply Chain Management*, 7–40. Springer.

Kim, J., and K. Rogers. 2005. "An Object-oriented Approach for Building a Flexible Supply Chain Model". *International Journal of Physical Distribution & Logistics Management* 35 (7): 481–502.

Lee, Y.-T. T., F. H. Riddick, and B. J. I. Johansson. 2011. "Core Manufacturing Simulation Data A Manufacturing Simulation Integration Standard: Overview and Case Studies". *International Journal of Computer Integrated Manufacturing* 24 (8): 689–709.

Lemaignan, S., A. Siadat, J. Y. Dantan, and A. Semenenko. 2006. "MASON: A Proposal for an Ontology of Manufacturing Domain". In *IEEE Workshop on Distributed Intelligent Systems: Collective Intelligence and Its Applications*, 195–200.

Narayanan, S., D. Bodner, U. Sreekanth, T. Govindaraj, L. McGinnis, and C. Mitchell. 1998. "Research in Object-oriented Manufacturing Simulations: An Assessment of the Ttate of the Art". *IIE Transactions* 30 (9): 795–810.

Ng, M., J. Park, and S. T. Waller. 2010. "A Hybrid Bilevel Model for the Optimal Shelter Assignment in Emergency Evacuations". *Computer-Aided Civil and Infrastructure Engineering* 25 (8): 547–556.

Osman, I. H. 1995. "Heuristics for the Generalised Assignment Problem: Simulated Annealing and Tabu Search Approaches". *Operations-Research-Spektrum* 17 (4): 211–225.

Pasupathy, R., and S. G. Henderson. 2011. "SimOpt: A Library of Simulation Optimization Problems". In *Proceedings of the 2011 Winter Simulation Conference*, edited by S. Jain, R. R. Creasey, J. Himmelspach, K. P. White, and M. Fu, 4075–4085. Piscataway, NJ, USA: IEEE.

Robinson, S. 2005. "Discrete-event Simulation: From the Pioneers to the Present, What Next?". *Journal of the Operational Research Society* 56 (6): 619–629.

Rosenblatt, M. J., Y. Roll, and D. Vered Zyser. 1993. "A Combined Optimization and Simulation Approach for Designing Automated Storage/Retrieval Systems". *IIE transactions* 25 (1): 40–50.

Rossetti, M., M. Miman, and V. Varghese. 2008. "An Object-oriented Framework for Simulating Supply Systems". *Journal of Simulation* 2 (2): 103–116.

Sobieszczanski-Sobieski, J., and R. T. Haftka. 1997. "Multidisciplinary Aerospace Design Optimization: Survey of Recent Developments". *Structural optimization* 14 (1): 1–23.

Sprock, T., and L. F. McGinnis. 2014. "Simulation Model Generation of Discrete Event Logistics Systems (DELS) Using Software Patterns". In *Proceedings of the 2014 Winter Simulation Conference*, edited by A. Tolk, S. D. Diallo, I. O. Ryzhov, L. Yilmaz, S. Buckley, and J. A. Miller. Piscataway, NJ, USA: IEEE Press.

Syarif, A., Y. Yun, and M. Gen. 2002. "Study on Multi-stage Logistic Chain Network: A Spanning Tree-based Genetic Algorithm Approach". *Computers & Industrial Engineering* 43 (1): 299–314.

Thiers, G. 2014. *A Model-Based Systems Engineering Methodology to Make Engineering Analysis of Discrete-Event Logistics Systems More Cost-Accessible*. Ph. D. thesis, Georgia Institute of Technology, Atlanta, GA.

Truong, T. H., and F. Azadivar. 2005. "Optimal Design Methodologies for Configuration of Supply Chains". *International Journal of Production Research* 43 (11): 2217–2236.

Xu, J., B. L. Nelson, and J. Hong. 2010. "Industrial Strength COMPASS: A Comprehensive Algorithm and Software for Optimization via Simulation". *ACM Transactions on Modeling and Computer Simulation (TOMACS)* 20 (1): 3.

Yang, T., Y. Kuo, and I. Chang. 2004. "Tabu-search simulation Optimization Approach for Flow-shop Scheduling with Multiple Processors - A Case Study". *International Journal of Production Research* 42 (19): 4015–4030.

Zhou, G., H. Min, and M. Gen. 2002. "The Balanced Allocation of Customers to Multiple Distribution Centers in the Supply Chain Network: A Genetic Algorithm Approach". *Computers & Industrial Engineering* 43 (1): 251–261.

AUTHOR BIOGRAPHIES

TIMOTHY SPROCK is a graduate student in the Stewart School of Industrial and Systems Engineering at Georgia Tech. Before returning to school he was employed by The Boeing Company in Philadelphia, PA and received a bachelors degree in applied physics and economics from Rensselaer Polytechnic Institute, Troy, NY. His email address is tsprock3@gatech.edu.

LEON F. MCGINNIS is Professor Emeritus in the Stewart School of Industrial and Systems Engineering at Georgia Tech and founder of the Keck Virtual Factory Lab. He is internationally known for his leadership in the material handling research community and his research in the area of discrete event logistics systems. A frequent speaker at international conferences, he has received several awards from professional societies for his innovative research, including the David F. Baker Award from IIE, the Reed-Apple Award from the Material Handling Education Foundation, and the Material Handling Innovation Pioneer award from Material Handling Management Magazine. He is author or editor of seven books and more than 110 technical publications. At Georgia Tech, Professor McGinnis has held leadership positions in a number of industry-focused centers and multi-disciplinary programs, including the Material Handling Research Center, the Computer Integrated Manufacturing Systems Program, the Sustainable Design and Manufacturing Program, and the Tennenbaum Institute for Enterprise Transformation. His current research explores the use of formal systems modeling methods to support systems engineering of discrete event logistics systems. Professor McGinnis is a Fellow of the Institute of Industrial Engineering. His e-mail address is leon.mcginnis@gatech.edu.