Proceedings of the 2025 Winter Simulation Conference
E. Azar, A. Djanatliev, A. Harper, C. Kogler, V. Ramamohan, A. Anagnostou, and S. J. E. Taylor, eds.

CONSTROBE - CONSTRUCTION OPERATIONS SIMULATION FOR TIME AND RESOURCE BASED EVALUATIONS

Joseph Louis[1]

[1]School of Civil and Construction Eng., Oregon State University, Corvallis, OR, USA

ABSTRACT

This paper introduces ConStrobe – Construction Operations Simulation for Time and Resource Based Evaluations – which is a simulation software that builds upon knowledge in construction field operations simulation by providing the capabilities of running High Level Architecture (HLA)-compliant distributed simulations and being amenable to automation from external programs written in the Python language for two-way communication with external data sources. These features are provided to overcome some of the major limitations of existing construction operations simulation tools that have hindered their widespread adoption by industry. The framework of this software is explained along with a sample demonstration case to provide users with an overview of its capabilities and understanding of its working. It is anticipated that the novel capabilities of ConStrobe can reduce the time and effort required to create simulations to enable process analysis for decision-making under uncertainty for complex operations in the construction and built operations domain.

1 INTRODUCTION

Construction engineering researchers (Abourizk 2010, Martinez 1996) have widely recognized simulation, and particularly discrete event simulation (DES), to be a powerful and suitable method for analyzing construction operations that are characterized by resource interdependencies, complex activity startup conditions, and uncertainties in activity durations and material quantities. The technical capabilities of DES thus provide a valuable means of estimating performance metrics for construction operations without needing to actually do them in the real-world (which is expensive and infeasible) or rely on mathematical formulas from the operations research domain (which make simplifying assumptions). Such performance measures include operation completion time, unit costs, equipment idle times etc. While Martinez and Ioannou (1999) identified the activity-scanning approach as the discrete event simulation paradigm of choice (over process-interaction) based on the criteria of application breadth, modeling paradigm, and flexibility for construction operations, Abourizk and Hague (2009) noted that process-interaction has been the traditional prevalent approach for construction operations simulation. Both methods have been implemented in simulation software, most notably CYCLONE (Halpin 1977), STROBOSCOPE (Martinez 1996), and Simphony (Hajjar and Abourizk 1999). While STROBOSCOPE and CYCLONE are general purpose simulation engines for construction, Simphony provides templates that can be used to create special purpose simulation tools for construction, (including a CYCLONE template for creating general purpose models) (Abourizk 2010).

These and other simulation engines for construction have collectively been used to model a wide range of construction operations including tunnel boring (Ruwanpura et al. 2001), earthmoving (Smith et al. 1995), pipe-spool module assembly (Mohammed et al. 2007), modular construction (Abiri et al. 2019), and disaster recovery (Louis et al. 2018), and asphalt paving operations (Mostafavi et al. 2012), among others. These DES models adopt a production view of construction operations and inform tactical decision-making related to process design, in contrast to the higher-level project view that primarily emphasizes scheduling and product completion. Abourizk (2010) characterizes the evolution of simulation tools in construction as having occurred in three stages – stage 1 being the advent of construction simulation in the 1970s, stage 2 being the use of object-oriented programming for simulation which focused on enhancing the modeling power and flexibility of construction simulation, and stage 3 which focused on the integration of simulation with other tools for visualizing operations and for conducting hybrid simulations.

Despite the advantages that simulation could offer for operations planning, it has not found adoption by practitioners in the construction industry. This situation has been investigated extensively, using both focus groups and surveys among researchers and industry practitioners (Lee et al. 2013, Leite et al. 2016) and survey of literature (Abdelmegid et al. 2020). These research efforts have identified several obstacles that hinder industry adoption of simulation including the nature of construction operations, the prevailing culture in the industry, preparedness and knowledge of construction workforce, and limitations of simulation tools themselves.

2 POINT OF DEPARTURE

This paper addresses limitations in construction simulation tools that can hinder their application to increasingly complex projects. It focuses on the difficulty of reusing DES models across different scenarios due to a lack of support for parameter customization based on existing project data sources. These issues can create barriers to the rapid development and deployment of simulation-based planning for real-world construction projects. One approach to overcoming these limitations is through the IEEE 1516 High Level Architecture (HLA), which provides a standard for creating distributed, time-synchronized simulation systems. HLA allows larger complex models to be modularized into smaller, reusable models that can be configured for different project needs.

Abourizk (2010) identified HLA as a promising direction for construction simulation and developed the Construction Synthetic Environment (COSYE) Framework, which has been implemented for the Simphony simulation system. Thus, while Simphony provides an integrated HLA-based modeling environment, similar capabilities are not currently available within the STROBOSCOPE framework. ConStrobe seeks to address this gap by offering a simulation platform inspired by STROBOSCOPE's modeling paradigm, while also enabling HLA compliance. The aim is to preserve the modeling flexibility and familiarity of STROBOSCOPE while supporting integration with distributed simulation systems through HLA. Joining an HLA *federation* – a collection of individual simulation *federates* that interact through a *Runtime Infrastructure (RTI)* – requires each federate to be HLA-compliant. ConStrobe thus provides HLA-compliant DES software for construction operations, that builds upon the capabilities of the STROBOSCOPE simulation language. It is explicitly clarified here that while ConStrobe is inspired by and uses much of the same modeling elements and paradigms of STROBOSCOPE, it has been developed independently by the author of this paper.

In addition to providing HLA capabilities for running distributed simulations, ConStrobe also provides users with a Python library (pyconstrobe) that can be used to integrate ConStrobe DES models with any data source that can be accessed by the Python programming language including GIS and BIM files, which can provide valuable spatial and product information to inform construction process analysis.

These two capabilities – being HLA-compliant for distributed simulations and being amenable to automation from Python – are the points of departure of ConStrobe from STROBOSCOPE. The remainder of this paper describes these capabilities in detail, along with a brief overview of the software platform and an example demonstrating interaction with GIS data sources.

3 OVERVIEW OF CONSTROBE

This section will consist of three subsections that provide a description of the ConStrobe simulation software, ConStrobe's HLA implementation, and its Python library implementation respectively.

3.1 ConStrobe Simulation Software

ConStrobe (Construction Operations Simulation for Time and Resource Based Evaluations) is a simulation software for construction operations simulation developed by Dr. Joseph Louis. ConStrobe is a general-purpose simulation engine that implements the three-phase activity scanning (AS) simulation strategy for processing discrete event simulation models that are represented as activity cycle diagrams (ACDs). While the details of the three-phase AS simulation algorithm are beyond the scope of this paper and can be found in Martinez (1996), the elements that make up activity cycle diagrams are described in Table 1.

Table 1: ACD elements in ConStrobe.

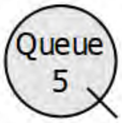
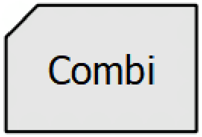
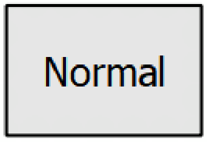
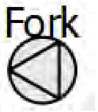
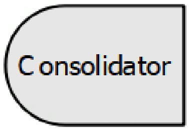
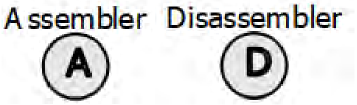
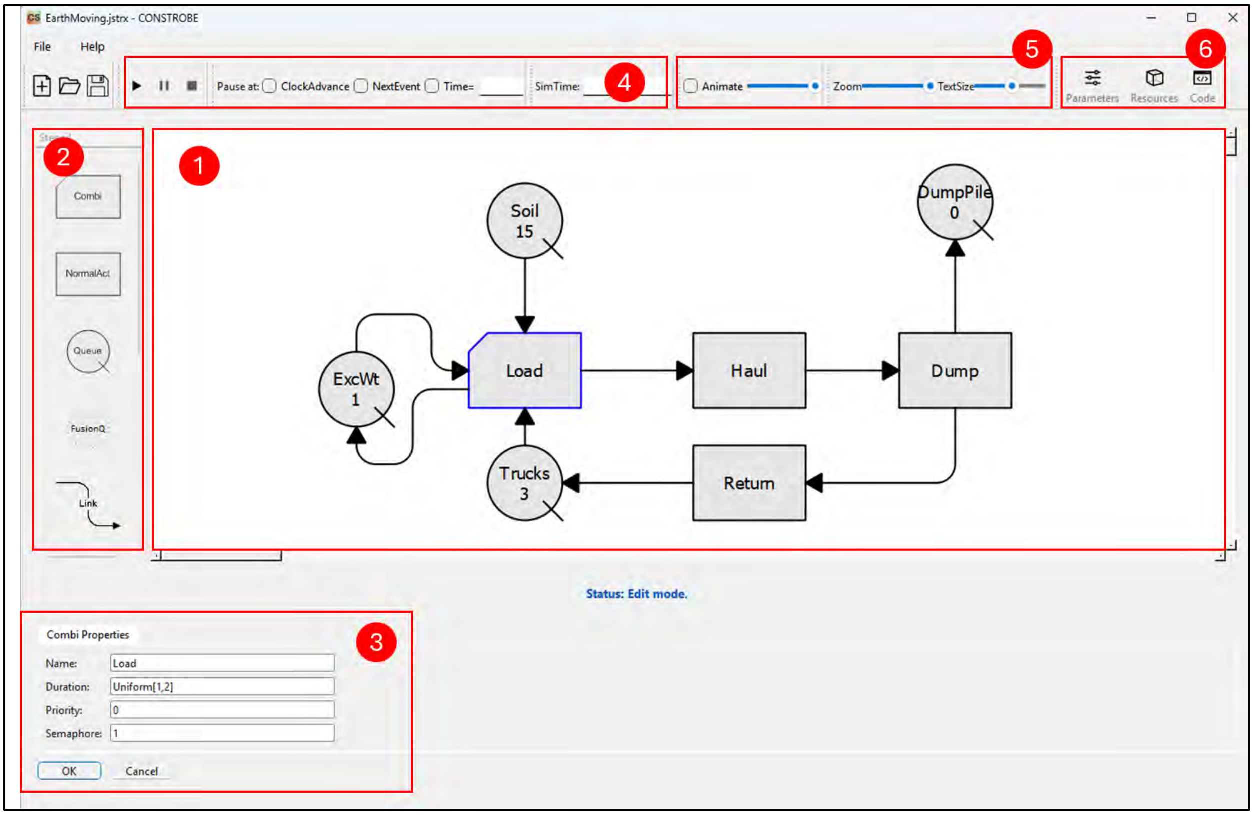
ACD Element Symbol	**Description**
	Queues are nodes that hold idle resources of a certain type until the resource is drawn away by an activity (specifically a Combi activity). Resources in queues are in an idle state and waiting to perform an activity. Queues can only precede Combis but can be preceded by any other node (except a Queue).
	A Combi is a type of activity that can start only when certain conditions are met, as opposed to starting immediately upon finishing its preceding activity. A Combi is associated with a duration and represents tasks in the operation that are subjected to multiple constraints. A Combi can only be preceded by a Queue but can precede any other node (except a Combi).
	A Normal is a type of activity that starts as soon as its preceding activity concludes. A Normal is associated with a duration and represents tasks in the operation that are not subjected to multiple constraints. A Normal can only be preceded by any Activity (Combi or Normal) or Fork, but not a Queue. A Normal can precede any other node (except a Combi).
	A Fork is a routing element that can route a specific type of resource probabilistically through its outgoing branches based on their proportional strength. It is an auxiliary element and can precede any node in an ACD except for a Combi.
	A Consolidator is an element that collects resources of any type until a specific condition is met, upon which it releases all collected resources through their respective outgoing links. These can precede any node other than a Combi and succeed any element other than a Queue.
	Assembler and Disassembler are auxiliary elements that are used to combine resources into a specific characterized resource and to separate them when needed in the model. These can precede any node other than a Combi and succeed any element other than a Queue.
	Links are used to connect the nodes to each other to create an ACD networks. These provide channels for the flow of resources between nodes and can only provide this capability for their assigned resource type. Links can be classified as a Draw Link (if it connects a Queue to a Combi) Release Link (if it emanates from an Activity).

Figure 1 provides a screenshot of the ConStrobe GUI, which is implemented using the Qt API with the following elements numbered in the figure:

1. ACD Area: This is the main portion of the GUI where users drag and drop modeling elements to create the ACD model of the operation. The ACD in the figure is for an earthmoving operation.
2. ACD Stencil: This area contains the modeling elements that can be used to build the ACD.

3. ACD Element Properties: This area provides the means of viewing and editing the selected element's attributes. In this case, the selected element is the Load Combi – outlined in blue.
4. Simulation Controls: These controls allow the user to start, pause, and stop the simulation.
5. View Controls: These controls allow the user to control the appearance of the ACD model.
6. Parameters, Resources, and Code: These three elements enable users to set parameters for the simulation, define resource types and properties, and write code to enhance ACD behavior.



Pause at: □ ClockAdvance □ NextEvent □ Time= SimTime:

□ Animate Zoom TextSize

Status: Edit mode.

Combi Properties

Name: Load

Duration: Uniform[1,2]

Priority: 0

Semaphore: 1

Figure 1: Screenshot of GUI in ConStrobe application.

The GUI provides the means of both building the ACD using a drag and drop interface and running the ACD to obtain statistical simulation results.

3.2 ConStrobe HLA Implementation

The key advancement that ConStrobe provides for construction operations simulation is its ability to join a federation of simulations as an HLA-compliant simulation federate. This is enabled through the integration of ConStrobe with the CERTI HLA framework (Noulard et al. 2009), which is an open-source HLA Run-Time Infrastructure (RTI) that implements the IEEE1516 Standard and supports the HLA1.3 specification fully, with partial support for the IEEE1516-v2000 and IEEE 1516-v2010 specifications. ConStrobe specifically adopts the IEEE 1516-2000 specification in its implementation.

As a result, any ConStrobe ACD model can join an existing federation as a simulation federate. Other federates that are connected to the federation can include other ConStrobe models or other HLA-compliant programs as illustrated in Figure 2. The light-blue shaded boxes represent HLA-compliant federate programs (which could include ConStrobe models), and the red arrows represent publishing data to the RTI, and the green arrows represent subscribing to data from the RTI.

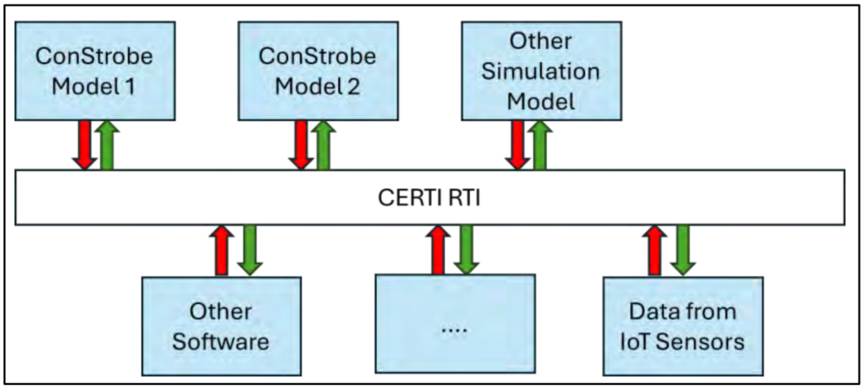


Figure 2: Schema for ConStrobe joining HLA federation as a federate.

As mentioned previously, one of the key capabilities enabled by HLA is that of distributed simulations. This means that individual separate ConStrobe models that have joined the HLA can be influenced by changes made in other simulations. In order to implement this functionality and to reduce ConStrobe modeler burden, we propose the schema shown in Figure 3, which includes a controller federate that subscribes to all messages published by any of the ConStrobe federates and then publishes pointed directives to specific ConStrobe federates. These directives can be one of the following commands:

1. **REMOVEFROMQUEUE QueueName Amount:** This command removes the requested *Amount* of resource from the queue named *QueueName*.
2. **ADDTOQUEUE QueueName Amount:** This command adds requested *Amount* of resource to the queue named *QueueName*.
3. **ENDACTIVITY ActivityName InstanceNumber:** This command ends the activity instance of type *Activity* with identifying number *InstanceNumber* from the Future Events List (Martinez 1996) to enable further processing of the model.

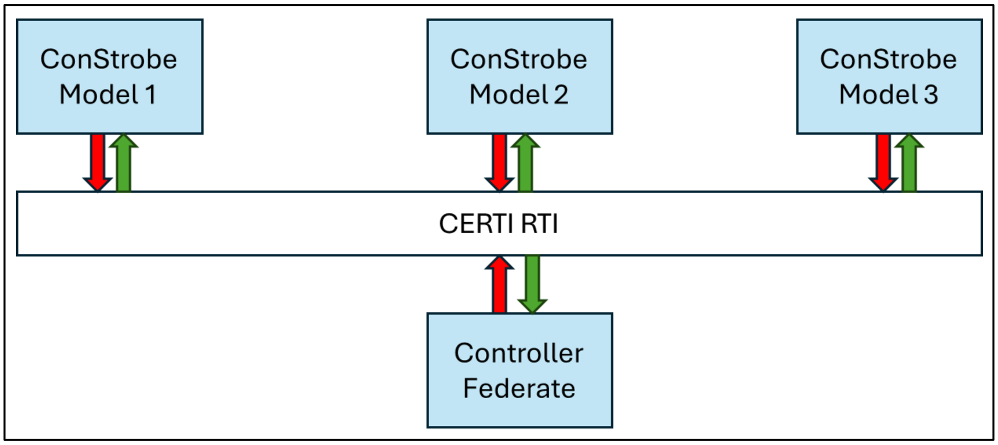


Figure 3: Schema for distributed simulations with ConStrobe.

The advantage of using a single controller federate to direct messages is that it eliminates the need for the ConStrobe modeler to manage incoming messages from other federates. This allows them to focus solely on modeling the intended behavior of the ACD. It must also be mentioned that additional non-ConStrobe federates can still join the schema in Figure 3.

3.3 ConStrobe Data Integration and Automation with Python

While the HLA schemas developed in the previous section would enable ConStrobe models to integrate seamlessly with other HLA-compliant federates and, by extension, a broad range of data sources that can inform operations simulation, it is also desirable to have a lightweight means of automating model simulation and data integration for cases where distributed simulation itself is not needed. Towards this end, ConStrobe provides the capability of being amenable to Python scripts that can extend the functionality of the native application by enabling access to any data source or capability that is provided by the Python programming language. This simple feature enables any ACD model of ConStrobe to be automatically opened, parameterized, and run iteratively, and have its results processed in any manner by Python programs. This enables the opportunity for the user to call the simulation application with any specific ACD as a function for performing operations analysis within the larger goal that they might have. This schema is shown in Figure 4.

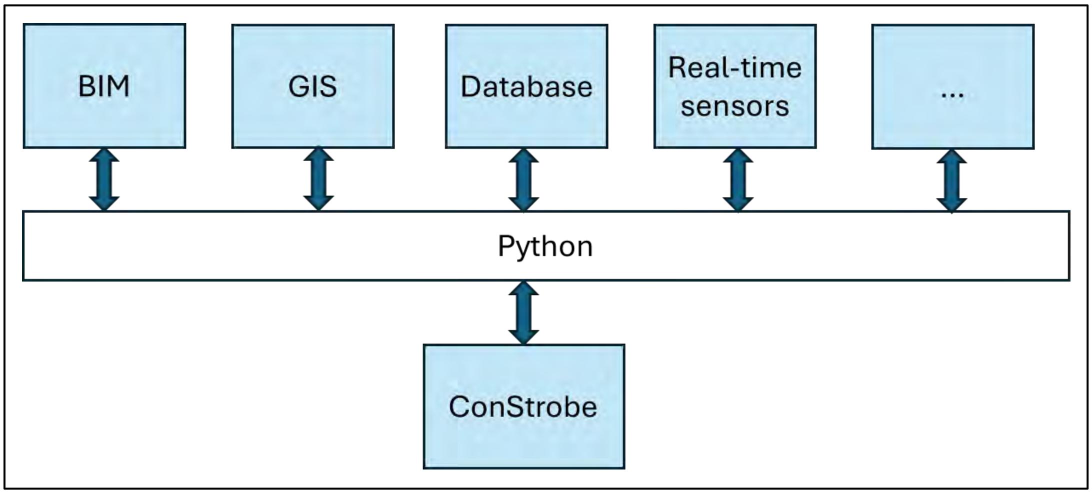


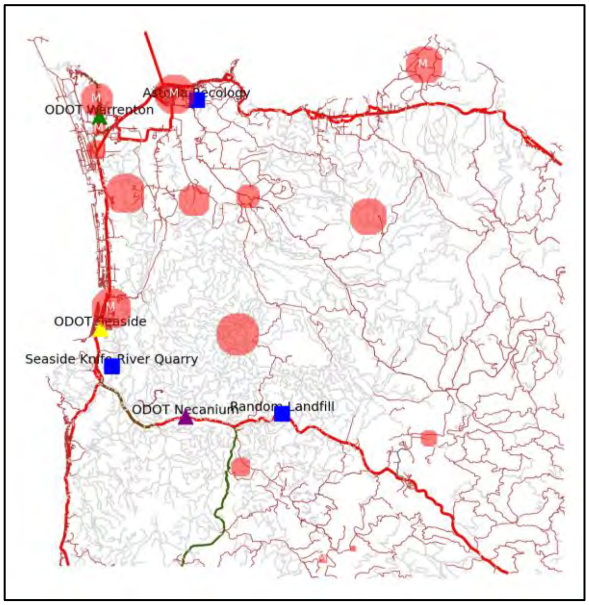
Figure 4: Schema for automating ConStrobe using Python.

Thus, not only can ConStrobe run independently to generate simulation results as described in Section 3.1 and be run as part of a simulation federation as described in Section 3.2; it can also be automated by Python as described here. The next section demonstrates this capability by using a generic earthmoving model to be populated by a GIS map containing roadways, depots, and landfills needed for a multi landslide recovery.

4 DEMONSTRATION OF PYTHON-ENABLED INTEGRATED GIS-DES SIMULATION

This example demonstrates how the model can be populated using data from an external source (GIS in this case) to solve the problem of determining the total recovery time after a series of landslides happen along roadways due to a large earthquake in the region. The user is tasked with evaluating the total time for recovery from multiple disasters (in this case, multiple landslides) based on the resources allocated to various depots in the region. This example assumes that a single type of operation (similar to earthmoving) is needed to remove landslide debris from the road and dispose of in pre-determined landfill sites. Thus, a distributed simulation is not required as all the landslides are assumed to have happened at the same time, and it is further assumed that depots do not share resources in this case. Therefore, a distributed simulation is not needed, but it is required to obtain spatial information regarding shortest routes to landfill sites from landslides to determine travel and return times. Therefore, we use Python to integrate GIS data for use in our ConStrobe simulation model, as described below. The python program first loads the GIS map with the roadways, depot, and landfill data, as shown in Figure 5.

Next policies in place are reviewed to determine which depots will handle which landslide and in what order. This is typically a function of the importance of the road to be repaired (clearing debris from landslides) as well as connectivity. Once these priorities are in place, the ConStrobe models are called for each depot with the sequence of landslides to clear. It is important to note that the duration of the haul and return activities are obtained from the shortest route between the landslide and the landfill for each landslide, shown in Figure 6.



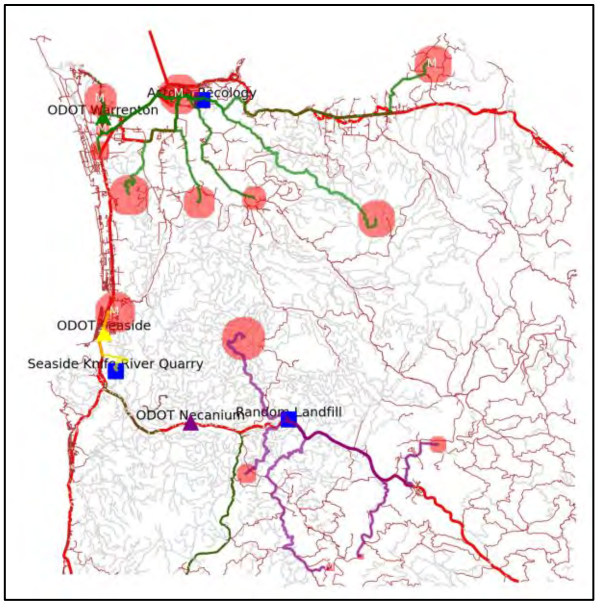


Figure 5: GIS data showing roads, depots, and landfill locations along with landslide locations with circle diameters representing debris volume.

Figure 6: Map showing the routes between landslides and landfills, which is used to populate simulation activity durations.

The results can be displayed in the form of a Gantt chart that can be easily interpreted by decision-makers as shown in Figure 7. This example shows that the simulation can be successfully combined with other data representations to provide the type of information that is required by the user.

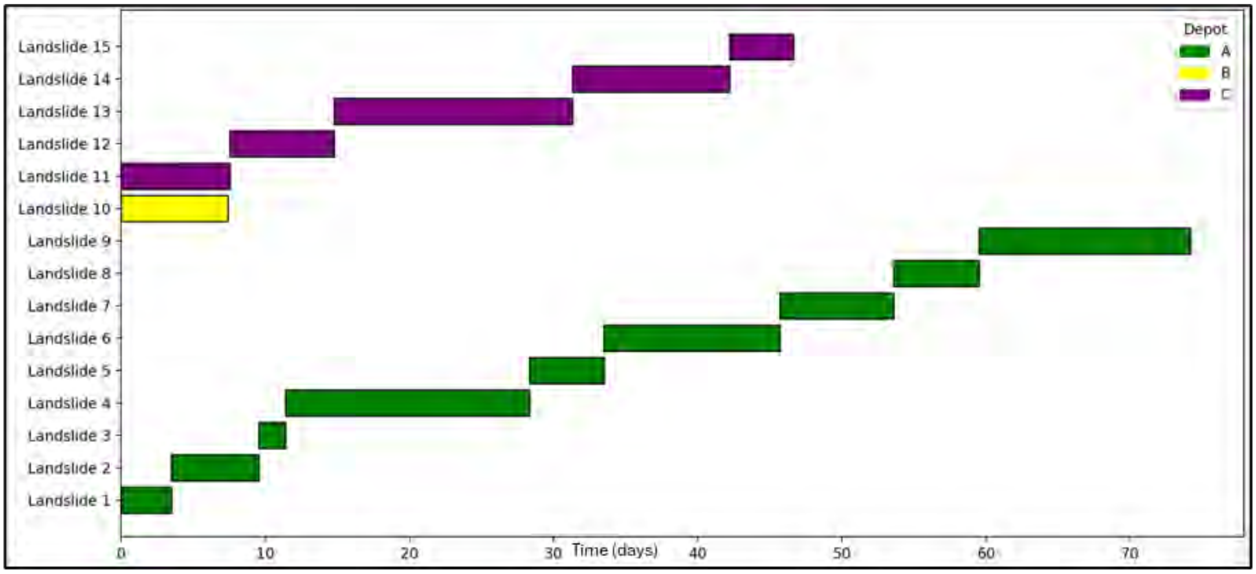


Figure 7: Gantt chart showing overall debris clearance time for all landslides.

5 CONCLUSIONS

This paper introduces the ConStrobe simulation software that has the following three major functionalities: simulation of ACD networks using the three-phase activity scanning approach, ability to join HLA federation of simulations, ability to be controlled by Python. The core elements of the GUI were described along with the type of elements needed to build a network. The example of an earthmoving operation was shown to demonstrate the modeling capabilities of ConStrobe, while an example of clearing multiple landslides was shown to demonstrate its working with Python and GIS data. These examples only serve to provide a general introduction to the capabilities of the software, and future work will provide more detailed tutorials for users to leverage these capabilities for their analysis.

REFERENCES

Abdelmegid, M. A., V. A. González, M. Poshdar, M. O'Sullivan, C. G. Walker, and F. Ying. 2020. “Barriers to Adopting Simulation Modelling in Construction Industry”. *Automation in construction* 111, 103046.

Abiri, B., J. Louis, and M. Riggio. 2019. “Decision-making for Cross-Laminated Timber Modular Construction Logistics using Discrete Event Simulation”. In *Modular and Offsite Construction (MOC) Summit Proceedings*, edited by M. Al-Hussein, 381-388. Banff, Alberta, Canada.

AbouRizk, S. 2010. “Role of Simulation in Construction Engineering and Management”. *Journal of Construction Engineering and Management* 136 (10) 1140–1153.

AbouRizk, S. M., and S. Hague. 2009. “An Overview of the COSYE Environment for Construction Simulation”. In *2009 Winter Simulation Conference (WSC),* 2624-2634 https://doi.org/10.1109/WSC.2009.5429307.

Hajjar, D. and S. AbouRizk. 1999. "Simphony: An Environment for Building Special Purpose Construction Simulation Tools." In 199*9 Winter Simulation Conference (WSC),* 998-1006 https://doi.org/10.1145/324898.324981.

Halpin, D. W. 1977. “CYCLONE–Method for Modeling Job Site Processes”. *Journal of the Construction Division* 103(3) 489-499.

Lee, S., A. Behzadan, A. Kandil, and Y. Mohamed. 2013. “Grand Challenges in Simulation for the Architecture, Engineering, Construction, and Facility Management Industries”. In *Proceedings of the 2013 ASCE International Workshop on Computing in Civil Engineering,* edited by I. Brilakis, S. Lee, B. Becerik-Gerber, 773-785. Virginia: American Society of Civil Engineers.

Leite, F., Y. Cho, A. H. Behzadan, S. Lee, S. Choe, Y. Fang, R. Akhavian, and S. Hwang. 2016. “Visualization, Information Modeling, and Simulation: Grand Challenges in the Construction Industry”. *Journal of Computing in Civil Engineering* 30(6), p.04016035.

Louis, J., A. Vijay, H. Wang, and D. Cox. 2022. “Evaluating the Impact of Equipment Selection on Debris Removal and Dependent Lifeline Infrastructure Recovery”. In Lifelines 2022: Advancing Lifeline Engineering for Community Resilience, edited by C. A. Davis, K. Yu, E. Taciroglu, 938-947. Los Angeles: American Society of Civil Engineers.

Martinez, J. C. 1996. *STROBOSCOPE: State and Resource Based Simulation of Construction Processes*, Ann Arbor: The University of Michigan.

Mohamed, Y., D. Borrego, L. Francisco, M. Al-Hussein, and S. AbouRizk. 2007. “Simulation-Based Scheduling of Module Assembly Yards: Case Study.” *Engineering, Construction and Architectural Management* 14 (3), 293–311.

Mostafavi, A., V. Valentin, D. M. Abraham, & J. Louis. 2012. “Assessment of the Productivity of Nighttime Asphalt Paving Operations”. *Journal of Construction Engineering and Management* 138(12), 1421-1432.

Noulard, E., J. Y. Rousselot, and P. Siron. 2009. “CERTI, an Open Source RTI, Why and How”. In *Spring Simulation Interoperability Workshop*, edited by Simulation Interoperability Standards Organization, 23-27. New York: Curran.

Ruwanpura, J. Y., S. M. AbouRizk, K. Er, and S. Fernando. 2001. "Special Purpose Simulation Templates for Tunnel Construction Operations." *Canadian Journal of Civil Engineering* 28(2), 222-237.

Smith, S. D., J. R. Osborne, and M. C. Forde. 1995. "Analysis of Earth-Moving Systems using Discrete Event Simulation." *Journal of Construction Engineering and Management* 121(4), 388-396.

AUTHOR BIOGRAPHIES

JOSEPH LOUIS is an Associate Professor in the School of Civil and Construction Engineering at Oregon State University. He received his Ph.D. degree and his M.Sc. in Civil Engineering from Purdue University in 2016 and 2010, respectively. He has created the ConStrobe Simulation Engine, and applies it to simulate and analyze various problems in the construction and built environment operations domains by integrating it with data sources such as Building Information Models (BIM), Geographic Information Systems (GIS), and real-time sensor data from construction equipment. His email address is joseph.louis@oregonstate.edu and his website, which contains further information about the ConStrobe software is https://josephlouis.org.