Proceedings of the 2021 Winter Simulation Conference
S. Kim, B. Feng, K. Smith, S. Masoud, Z. Zheng, C. Szabo and M. Loper, eds.

DYNAMIC, DATA-DRIVEN SIMULATION IN CONSTRUCTION USING ADVANCED METADATA STRUCTURES AND BAYESIAN INFERENCE

Ramzi Roy Labban
Stephen Hague
Elyar Pourrahimian
Simaan AbouRizk

University of Alberta
5-080 Natural Resources and Engineering Facility
Department of Civil and Environmental Engineering
Edmonton, AB T6G 2W2, CANADA

ABSTRACT

Effective project control in construction requires the rapid identification and subsequent mitigation of deviations from planned baselines and schedules. Although simulation has been used to successfully plan projects in the pre-construction phase, the use of simulation for project control during execution remains limited. Current real-time simulation strategies have difficulty self-adapting in response to deviations from planned baselines, requiring experienced simulation experts to manually update the input parameters of simulation models. This study is proposing a dynamic, data-driven simulation environment that is capable of minimizing the manual intervention required to incorporate as-built construction data in real-time by coupling newly-developed metadata structures with Bayesian inference. Still in development, an overview of the proposed simulation environment is presented, details of the advanced data structures are discussed, and preliminary functionality of the environment is demonstrated.

1 INTRODUCTION

To ensure profitability, construction organizations must continuously refine project plans, ensuring that resource usage is optimized, project durations are minimized, and potential risks are mitigated (Halpin and Riggs 1992). For over 40 years, construction researchers have used simulation to model construction operations for improved project management (Halpin 1973). While this has resulted in a large collection of simulation-based research, relatively few of these academic contributions have been successfully implemented for project control in the execution phase of construction (AbouRizk 2010; Leite et al. 2016).

Two factors limiting the use of simulation beyond the planning stages in construction are (1) the rigid nature of simulation models (Lee et al. 2013) and (2) their inability to appropriately integrate frequent and transient change (Lugaresi and Matta 2018). Construction projects are variable in nature, with as-built data regularly deviating from planned baselines and schedules. In contrast to other industries, incorporation of as-built construction data into simulation models often necessitates model recalibration each time new data are integrated. Both time-consuming and requiring expert knowledge, current real-time simulation strategies are not able to generate results in the time-window required to be effectively leveraged for construction project control (AbouRizk 2010).

The development of self-adaptive simulation models that are capable of incorporating as-built data in real-time are required before simulation can be effectively used during project execution (Lugaresi and Matta 2018; Lee et al. 2013). As a step towards this goal, this research is proposing an innovative simulation environment capable of dynamically and automatically updating simulation models with real-time, as-built information. The proposed environment uses advanced metadata structures to achieve data-driven self-adaptation and applies Bayesian inference to more appropriately integrate the frequent deviations and atypical outliers characteristic of construction. This newly-developed approach has the potential to substantially reduce the manual intervention and expert knowledge required for real-time model updating, thereby overcoming existing challenges limiting the implementation of simulation in construction practice. Still in the preliminary stage of development, the aim of this paper is to provide an overview of the architecture and data flow of the proposed simulation environment and to detail the innovative metadata structures of the modeling database. Preliminary functionality of the system is demonstrated through an illustrative example.

2 BACKGROUND

Project control involves the "planning, monitoring, and controlling" of project execution in an attempt to mitigate the negative consequences of deviations from planned project baselines (CII 2021). Many projects in construction are sequential in nature, requiring the successive completion of multiple tasks by various crews (i.e., resources). In contrast to more iterative construction projects, such as earthmoving, where project deviations generally do not affect system flow, deviations in sequential construction often result in changes to the underlying logic of the system. Incorporating real-time data into simulation models of sequential construction operations, therefore, often requires manual intervention and model recalibration.

For simulation to be effective in the execution phase of construction projects, strategies allowing simulation models to be automatically paused, updated with as-built data, and restarted, are required. An idealized simulation approach is illustrated in Figure 1. Here, the progress of a project is simulated from Point A to Point C. Once execution begins, the simulation model is updated with as-built project data. Point B represents an arbitrary point along the lifecycle where an update to the parameters and/or topology of the model takes place. The ideal simulation strategy allows the simulation to continue from Point B to Point C based on real-time progress of the project from Point A to Point B.

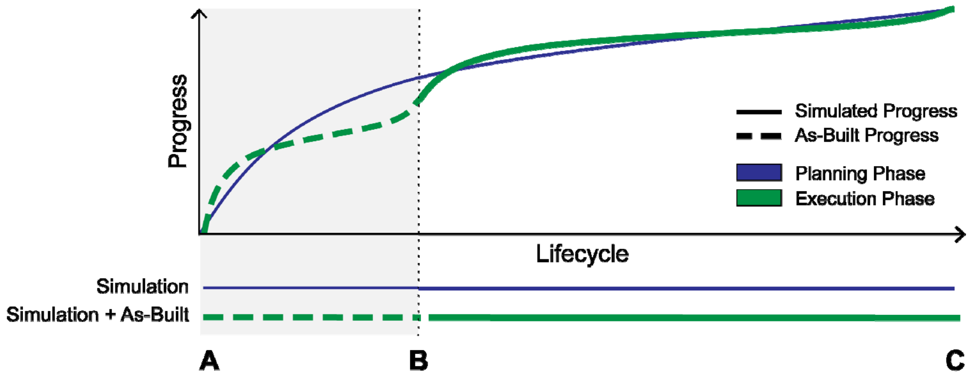


Figure 1: Simulation lifecycle during planning and execution phases of a construction project.

3 LITERATURE REVIEW

3.1 Dynamic, Real-Time Simulation in Construction

Real-time simulation (RTS) has the potential to improve the accuracy of performance forecasting over traditional simulation approaches. Several researchers have adopted RTS in construction, including approaches for generating new motion plans to prevent crane collisions (Zhang et al. 2012), improving input models in tunneling projects (Chung et al. 2006), and updating activity durations during the construction of a concrete plant (Lu et al. 2007). Although successful at incorporating as-built project data, many RTS approaches in construction have been limited to updating parameter values.

Even fewer attempts to develop self-adaptive simulation models in construction have been made. In 2008, Song, Ramos, and Arnold proposed a RTS framework for modeling heavy construction operations (Song et al. 2008). Data and process knowledge were used together to enable a self-adaptive modeling process. However, the modeling approach proposed by Song et al. cannot be paused and adjusted with real-time data (i.e., model cannot be paused at Point B in Figure 1). Rather, the model is run, real-time data are input, and the complete model is then re-run from the start (i.e., Point A). Furthermore, the approach does not appear to be generalizable, necessitating an original, customized model to be created for each intended application.

3.2 Research Gap

Several issues preventing RTS implementation, namely data collection, automated inputs, auto-validation, initialization, synchronization, model generation, reactiveness, shared information, modeling methods, and interfaces, have yet to be addressed in construction (Lugaresi and Matta 2018). Current approaches do not consider issues related to data streams, dataset size, and different subsets of data that may be needed (Waschneck et al. 2016).

Many RTS approaches in construction are limited to parameter updating. Where self-adaptive modeling has been attempted, a lack of generalizability negates potential benefits. In the aforementioned case, manual intervention is still required to develop a customized model, which is often beyond the expertise of practitioners. As such, RTS remains limited to efficient look-ahead scheduling, progress monitoring, and productivity measurement. A generalizable environment capable of automatically (1) generating and (2) updating both the parameters and topology of simulation models would transform the way simulation is used for project management and control in construction practice.

Integrating advanced metadata structures with Bayesian inference could address two of the primary factors limiting RTS in construction—specifically adaptability and reactiveness. Advanced metadata structures could minimize the intervention required to execute parameter and topology changes to a simulation model, and combining this approach with Bayesian inference could more appropriately integrate frequent deviations and atypical outliers inherent to construction. However, a simulation environment capable of achieving these objectives has yet to be developed.

4 PROPOSED ENVIRONMENT

This research aims to overcome the limitations of previous RTS studies by coupling advanced metadata structures with Bayesian inference to develop a self-adaptive simulation environment that enables dynamic, data-driven simulation in construction. The simulation environment centers around a newly-developed generic database schema that supports model definitions for products, processes, and the environment for a wide variety of construction domains. The proposed environment also encompasses several supporting components, including the functions, algorithms, and human interfaces that:

1. Build a simulation model from data.
2. Run the model and generate associated artificial histories of simulation runs.
3. Capture and integrate as-built data at desired intervals (i.e., Point B in Figure 1).
4. Update model parameters, including product status, resource availability, crew composition, and resource productivity.

An overview of the proposed architecture is illustrated in Figure 2. In construction, the input data used to define the simulation model are often stored across multiple data sources, including (1) static data, which are often stored as design drawings, building information models, design specifications, and estimates, and (2) dynamic data, such as time sheets, inventories, schedules, progress reports, and resource availability, which are typically stored as part of an organization's corporate systems.

Input data are retrieved using data adaptors, which transform and feed required data into the modeling database. Then, data in the modeling database are read by a model generator, which is responsible for building the simulation model. Once built, the simulation model is run. An artificial history of the simulation scenario and corresponding model statistics are produced, and output data are written back into the modeling database after each simulation run. The modeling database receives dynamic input in the form of either manual or automated updates to its parameters as well as actual project data. To achieve real-time updating, simulation output data, prior historical data, initial assumptions of the model, and real-time project information are retrieved from the modeling database and integrated to generate a new set of performance parameter assumptions for subsequent runs. The modeling database can also act as a data source for optimization and/or visualization modules. An overview of the data flow is illustrated in Figure 3. Functionality of the various components are detailed as follows.

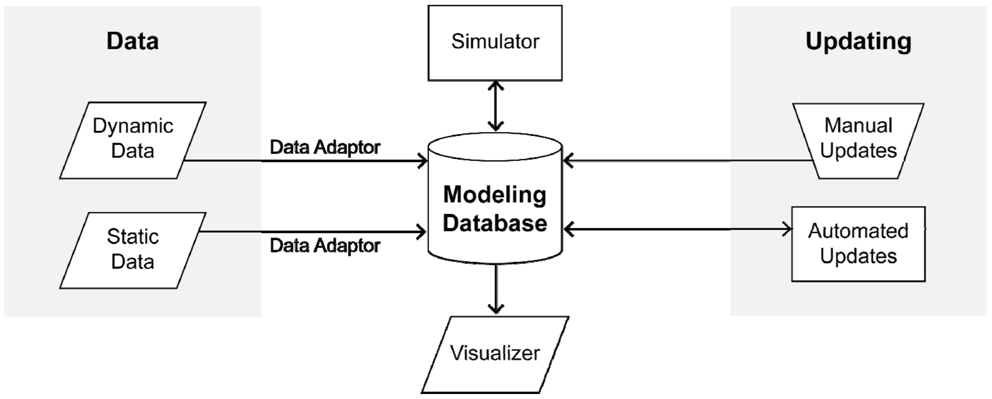


Figure 2: Architecture of proposed simulation environment.

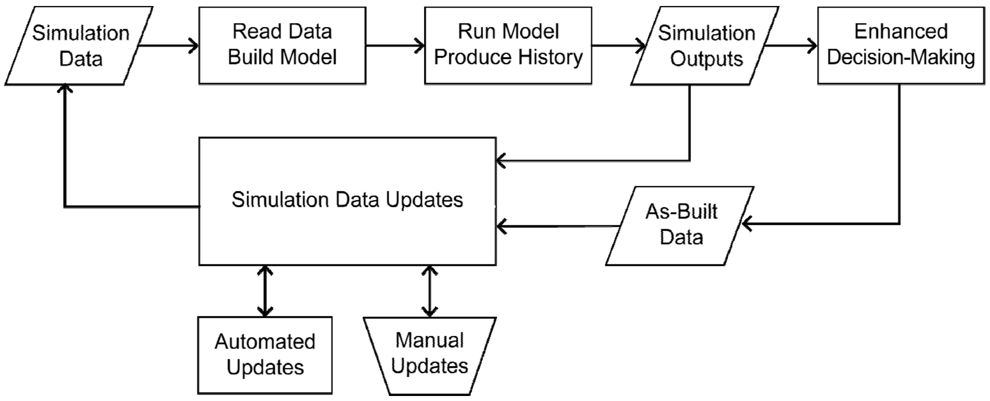


Figure 3: Data flow of proposed framework.

4.1 Advanced Metadata Structures

The metadata structures define the product, process, and environment components of the model. Product data define what the construction operation will produce, process data define the tasks and resources required to produce the product, and environment data define the construction environment. Tables record the date stamp of the record (i.e., VERSION field, where applicable), thereby permitting modifications to be made while maintaining previous historical versions. Data tables are detailed as follows.

4.1.1 Product Table

The *Product Input Table* (Table 1) defines the products of the construction operation and associates each product with a template of tasks required to build the product (Table 2). Products can be defined in a hierarchical manner, if required, by adding child products; there is no limit to the hierarchy level.

Table 1: Description of product table fields.

Field	**Description**
Product Input Table	
ID	Unique identifier
PRODUCT	Human-readable identifier for the product
QTY	Quantity of products of this type of construction
TEMPLATE	Human-readable identifier for the template
PARENT	Name of parent product for a child product, or null for a top-level product

4.1.2 Process Tables

Process tables consist of task-related tables and resource-related tables. Task-related tables (Table 2) include *Template* and *Sequence Input Tables*. The *Template Input Table* defines the tasks required to build a specific kind of product. Notably, multiple products can share the same template. Every task in the template defines the crew (i.e., resources) required to perform the task, the productivity rate of the crew, the product level at which the task is being performed, and the quantity coefficients used to calculate the amount of work required of the crew. The system supports both constant and stochastic productivity rates. The *Sequence Input Table* defines the sequence of the tasks in each of the templates. Sequences can have complex flows, including linear and parallel flows.

Table 2: Description of task-related table fields.

Field	**Description**
Template Input Table	
ID	Unique identifier
TEMPLATE	Human-readable identifier for the template
TASK	Human-readable identifier for the task
QTY_M	Quantity multiplier (≥ 1) used to convert the product quantity into the task quantity
QTY_K	Constant value (≥ 0) added to the task quantity regardless of the product quantity
RESOURCETYPE	Name of crew required to perform task
PRODUCTIVITY	Productivity rate (as a constant or probability distribution) of the crew performing the task; supports manual and/or automated updating
LEVEL	Product hierarchy level (e.g., 1, 2, 3, etc.) at which the task is performed
PERSIST	Flag indicating that resources are retained from the beginning of the first task to the end of the last task indicated in the sequence table, allowing resources to be carried across activities (e.g., a truck transporting material through multiple steps)
CUMULATIVE	Flag indicating that the task quantity should be calculated as the cumulative production rather than the product quantity (e.g., a truck whose traveling distance increases incrementally with each trip)
VERSION	Date stamp of record
Sequence Input Table	
ID	Unique identifier
TEMPLATE	Human-readable identifier for the template
PREDECESSOR	Name of predecessor task
SUCCESSOR	Name of successor task
VERSION	Date stamp of record

Resource-related tables (Table 3) include the *Crew Composition Input Table*, which defines the resources (e.g., labor and equipment) that encompass a crew, and the *Resource Availability Input Table*, which defines the availability of resources over time, permitting the ramp-up or ramp-down of resources through the lifecycle of the project.

Table 3: Description of resource-related table fields.

Field	**Description**
Crew Composition Input Table	
ID	Unique identifier
RESOURCETYPE	Name of crew required to perform task
RESOURCE	Name of resource
NUMBER	Quantity of resource required in crew
VERSION	Date stamp of record
Resource Availability Input Table	
ID	Unique identifier
RESOURCE	Name of resource
FROM	Start date of specified period for record, or null if record applies throughout model life
TO	End date of specified period for record, or null if record applied throughout model life
QTY	Quantity of resource available during specified period
VERSION	Date stamp of record

4.1.3 Environment Tables

Environment tables (Table 4) include the project *Calendar Input Table* and the *Setup Table*. The *Calendar Input Table* defines the working hours of a resource by date and/or time. The *Setup Table* contains the definitions of the global simulation parameters, including the random number seed, start date for the project, and the number of Monte Carlo simulation iterations to perform.

Table 4: Description of environment table fields.

Field	**Description**
Calendar Input Table	
ID	Unique identifier
CALENDAR	Name of calendar
RESOURCE	Name of resource
FROM	Start date of specified period for record, or null if record applies throughout model life
TO	End date of specified period for record, or null if record applies throughout model life
SHIFTFROM	Start time of shift
SHIFTTO	End time of shift
VERSION	Date stamp of record
Setup Table	
ID	Unique identifier
PARAMETER	Name of parameter
PARAMVALUE	Value of parameter

4.2 Model Generation and Execution

The model generator is executed using a customized, in-house developed software application that reads updated information from the input tables in the modeling database and constructs a corresponding Simphony General Template model (as a .sim file) (AbouRizk et al. 2016). An example is provided in Figure 4. The *Simphony* model is hierarchical and incorporates the components defined in the database, including crew availability and templates. The model is linked to the modeling database, allowing (1) products to be read during simulation execution and (2) simulation results to be written back into the database.

The resulting simulation model can be executed in one of two ways. First, the model can be loaded into the graphical user interface provided within *Simphony* (AbouRizk et al. 2016), namely the Simphony Modeling Element. Once loaded, the model may be viewed, edited (if desired), and executed. Alternatively, if automation is desired, the model can be executed by the customized, in-house developed software application prototype. In both situations, the simulation proceeds as follows. First, a connection to the central database is established, and the *Product Input Table* is read. Each top-level product is routed as an entity to the appropriate template sub-model. Next, the discrete-event simulation is performed, with entities representing products flowing through their respective sub-models. As the simulation proceeds, statistical information is generated and stored. Finally, after all products are complete, the connection to the database is re-established, and the simulation outputs are written into the output tables.

4.3 Model Outputs

Simulation output statistics are used both for decision-making and for simulation model updating. Outputs of the simulation model are collected and stored in output tables (Table 5). Statistics and metrics generated by the dynamic simulation model, as well as performance metrics of the actual operation, are used to assess performance of the model and the operation, respectively. Both model-generated and actual performance metrics are used to update the simulation, as required.

Table 5: Summary of output tables.

Output Table	**Description of Table Fields**
Task Time	Start and end times of each task for every product, amount of time waiting for resources to become available, and amount of time spent performing a task
Resource Use	Utilization level of individual resources over time, including the total and in-use quantities of resources at a given time
Crew Queuing	Queuing information of products for crews over time
Model Statistics	General statistics generated by the simulation environment

4.4 Model Updating

Statistics generated by the dynamic simulation model (Table 5) are collected and used to update parameters. Performance data and metrics from actual operations are stored in actual input tables. Currently, actual durations of tasks based on as-built data are stored in an *Actual Task Time Input Table*. Output data from the simulator are integrated with prior historical data, initial assumptions for the model, and as-built data to generate a new set of assumptions for subsequent simulation runs. Updates can be made manually or automatically.

Currently, updating is limited to resource-related data (Table 3), including changes to the *Crew Composition*, *Resource Availability*, and *Template* (specifically, the PRODUCTIVITY field) *Input Tables*, as well as updates to the project *Calendar Input Table* (Table 4). While model updating does not currently include updates to the process portion of the model definitions, the meta-model is structured in a manner that can allow model topology to be updated through the addition or removal of rows from the *Template Input Table* or the *Sequence Input Table* (Table 2).

4.4.1 Manual Updating

Manual updates can be performed by the end user at any given point in time through the updating of model parameters with new values, as required. The end user manages parameter updates through a specialized interface that allows for modification of model definition data mid-simulation lifecycle (i.e., Point B in Figure 1), where the simulation is stopped at a specified time, values are altered, and the simulation is continued. When manually updating the simulation, the system must ensure that the simulation continues with the actual project scope remaining after it is restarted. First, products are flagged with their completion status at each stopping point, ensuring that the simulation can restart at the stopping point. Actual project information that is input into the modeling database can result in changes to the remaining scope. As such, updates from real project information may include updates to both the parameters and scope.

4.4.2 Automated Updating

The simulation may also be updated using an automated approach. The updating component of the simulation environment uses Bayesian inference to integrate historical information with real-time, as-built data for updating purposes. At present, automated updating of only the productivity of a task is supported.

4.4.2.1 Bayesian Inference for Automated Parameter Updating

During automated updating, the productivity rate of a task is defined by a Bayesian distribution (PRODUCTIVITY field). The Bayesian distribution encompasses a probability distribution, such as a triangular, log-normal, or beta, together with a multivariate distribution, known as the prior distribution, that specifies parameter uncertainty (Ang and Tang 2006). When the model is initially executed, numerical integration is performed to construct a frequency polygon, using the approach previously-described by Hague and AbouRizk (2019), which approximates the prior predictive distribution. The value is sampled from the frequency polygon.

When new data are available, Bayesian inference is used to construct the posterior distribution for automated parameter updating. The posterior distribution is a multivariate distribution that combines the information contained in the prior distribution with the new data, thereby representing the revised uncertainty of the parameters. When the model is executed, numerical integration is again performed to construct the frequency polygon (Hague and AbouRizk 2019). This time, however, the frequency polygon will approximate the posterior predictive distribution. An updated productivity value is then sampled from the new frequency polygon.

5 FRAMEWORK APPLICATION PROTOYPE

An illustrative example of an industrial pipe spool fabrication application was developed to demonstrate the functionality of the proposed simulation approach. A Microsoft Access database containing the input/output tables described in Section 4 was created, and the input tables were populated using data adapted by the authors from a real project. A prototype implementing the proposed approach was developed in C# and the Microsoft Visual Studio IDE (Microsoft 2021). The Core Services component of the *Simphony* environment (AbouRizk et al. 2016) was used as the back-end simulation engine.

5.1 Model Definitions

Data were input into the modeling database as follows. The spools (NAME) to be produced, including their size (QTY), the steps required to produce this type of spool (TEMPLATE), and whether to deal with the spool at the spool or joint level (PARENT) were defined in the *Product Input Table*.

The various tasks (TASK) of the pipe spool fabrication process, as well as the associated quantities (QTY_M, QTY_K), the type of crew (RESOURCETYPE), the productivity rate (PRODUCTIVITY), the hierarchy level at which the task applies (LEVEL), and other supportive modeling flags were defined in the *Template Input Table*. Predecessor (PREDECESSOR) and successor (SUCCESSOR) tasks were defined in the *Sequence Input Table*.

Crews (CREW) and the quantity (NUMBER), dates of availability (FROM, TO), and shift dates (SHIFTFROM, SHIFTTO) of their associated resources (RESOURCE) were defined in the *Crew Composition*, *Resource Availability*, and *Calendar Input Tables*, respectively. Finally, the start date, time unit, seed, and run for the simulation model were defined in the *Setup Table*.

5.2 Simulation Model

The simulation model abstracted the pipe spool fabrication activities required to produce the pipe spool products. The fabrication process was defined as a simulation model using the data structures of the simulation environment. The Simphony Modeling Element of *Simphony* (AbouRizk et al. 2016) was used to visualize the resulting simulation model. A portion of the model is illustrated in Figure 4.

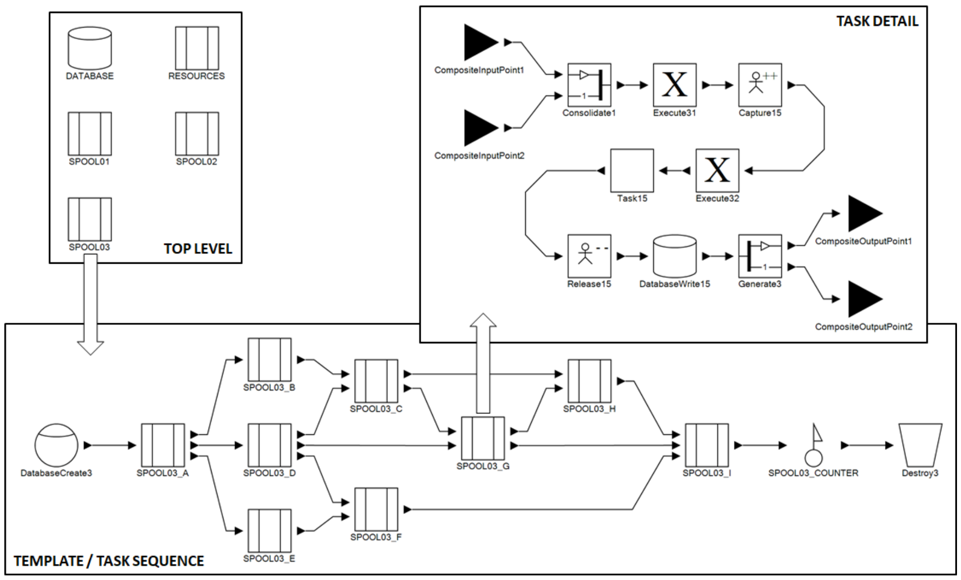
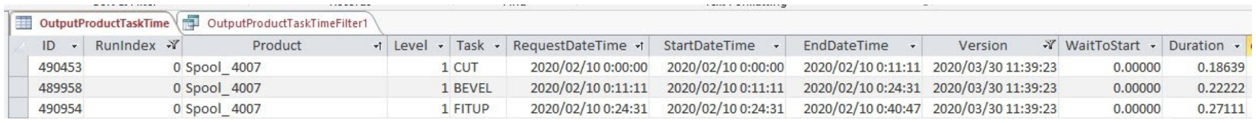


Figure 4: Portion of simulation model for illustrative example.

5.3 Model Outputs

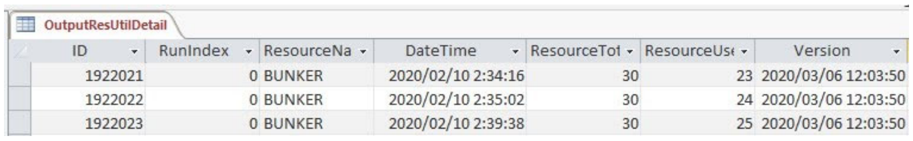
Outputs of the model were stored in the four output tables summarized in Table 5. Start (STARTDATETIME), end (ENDDATETIMES) times, amount of time waiting for resources to become available (WAITTOSTART), and the duration (DURATION) of each task (TASK) for each product (PRODUCT) were determined and written to the *Task Time Output Table* (Figure 5).



ID	RunIndex	Product	Level	Task	RequestDateTime	StartDateTime	EndDateTime	Version	WaitToStart	Duration
490453	0	Spool_4007	1	CUT	2020/02/10 0:00:00	2020/02/10 0:00:00	2020/02/10 0:11:11	2020/03/30 11:39:23	0.00000	0.18639
489958	0	Spool_4007	1	BEVEL	2020/02/10 0:11:11	2020/02/10 0:11:11	2020/02/10 0:24:31	2020/03/30 11:39:23	0.00000	0.22222
490954	0	Spool_4007	1	FITUP	2020/02/10 0:24:31	2020/02/10 0:24:31	2020/02/10 0:40:47	2020/03/30 11:39:23	0.00000	0.27111

Figure 5: Portion of *Task Time Output Table*.

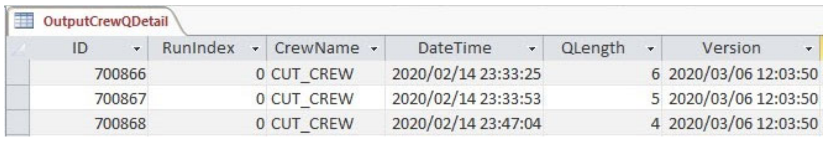
The total (RESOURCETOTAL) and in-use (RESOURCEUSE) quantities of resources (RESOURCETYPE) as well as their queuing information (QLENGTH) over time were output into the *Resource Use* (Figure 6) and *Crew Queuing* (Figure 7) *Output Tables*, respectively. A variety of general statistics, including standard deviation, productivity rates, maximum length, maximum utilization, and average inter-arrival, were calculated and output into the *Model Statistics Output Table*.



OutputResUtilDetail

ID	RunIndex	ResourceNa	DateTime	ResourceTot	ResourceUs	Version
1922021	0	BUNKER	2020/02/10 2:34:16	30	23	2020/03/06 12:03:50
1922022	0	BUNKER	2020/02/10 2:35:02	30	24	2020/03/06 12:03:50
1922023	0	BUNKER	2020/02/10 2:39:38	30	25	2020/03/06 12:03:50

Figure 6: Portion of *Resource Use Output Table*.



OutputCrewQDetail

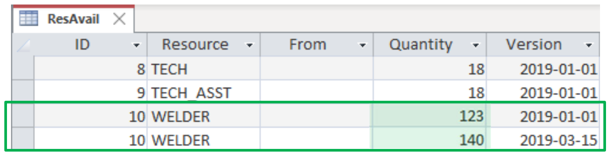
ID	RunIndex	CrewName	DateTime	QLength	Version
700866	0	CUT_CREW	2020/02/14 23:33:25	6	2020/03/06 12:03:50
700867	0	CUT_CREW	2020/02/14 23:33:53	5	2020/03/06 12:03:50
700868	0	CUT_CREW	2020/02/14 23:47:04	4	2020/03/06 12:03:50

Figure 7: Portion of *Crew Queuing Output Table*.

5.4 Model Updating

Updates to the model were performed both manually and in an automated manner. Manual updates included changes to resource availability, crew compositions, and crew productivity. To perform manual updates, the simulation run was paused at Point B, and model parameters were updated by changing values in the appropriate locations in the data structure tables. Updated values were derived from either (1) observations of simulation run results produced up to Point B or (2) assumptions designed to mimic expected occurrences in a real industrial setting. New records holding the updated values carried a date/time stamp (VERSION) to signify to the model which value to use and to enable traceability.

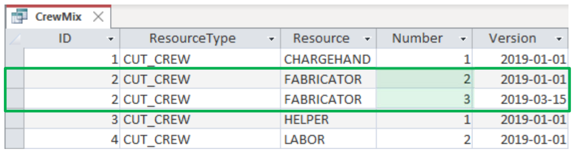
In the illustrative example, resource availability was changed at Point B to reflect the addition of a specific resource type (a welder) due to changes to the operation's resource availability plan (Figure 8a). Crew composition was also altered to reflect corrections required based on changes to actual crew compositions, which involved the addition of one fabricator to the Cut Crew (Figure 8b). Finally, the productivity of the FitUp Crew was also manually updated to represent an increase in the crew's productivity rate onsite (Figure 8c).



ResAvail

ID	Resource	From	Quantity	Version
8	TECH		18	2019-01-01
9	TECH_ASST		18	2019-01-01
10	WELDER		123	2019-01-01
10	WELDER		140	2019-03-15

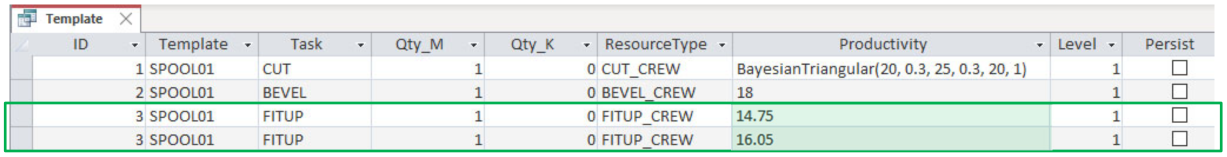
(a)



CrewMix

ID	ResourceType	Resource	Number	Version
1	CUT_CREW	CHARGEHAND	1	2019-01-01
2	CUT_CREW	FABRICATOR	2	2019-01-01
2	CUT_CREW	FABRICATOR	3	2019-03-15
3	CUT_CREW	HELPER	1	2019-01-01
4	CUT_CREW	LABOR	2	2019-01-01

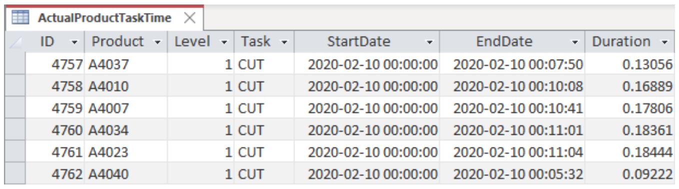
(b)



Template

ID	Template	Task	Qty_M	Qty_K	ResourceType	Productivity	Level	Persist
1	SPOOL01	CUT	1	0	CUT_CREW	BayesianTriangular(20, 0.3, 25, 0.3, 20, 1)	1	☐
2	SPOOL01	BEVEL	1	0	BEVEL_CREW	18	1	☐
3	SPOOL01	FITUP	1	0	FITUP_CREW	14.75	1	☐
3	SPOOL01	FITUP	1	0	FITUP_CREW	16.05	1	☐

(c)



ActualProductTaskTime

ID	Product	Level	Task	StartDate	EndDate	Duration
4757	A4037	1	CUT	2020-02-10 00:00:00	2020-02-10 00:07:50	0.13056
4758	A4010	1	CUT	2020-02-10 00:00:00	2020-02-10 00:10:08	0.16889
4759	A4007	1	CUT	2020-02-10 00:00:00	2020-02-10 00:10:41	0.17806
4760	A4034	1	CUT	2020-02-10 00:00:00	2020-02-10 00:11:01	0.18361
4761	A4023	1	CUT	2020-02-10 00:00:00	2020-02-10 00:11:04	0.18444
4762	A4040	1	CUT	2020-02-10 00:00:00	2020-02-10 00:05:32	0.09222

(d)

Figure 8: Portion of updated (a) *Resource Availability*, (b) *Crew Composition*, (c) *Template*, and (d) *Actual Task Time Input Tables*, with manual updates (green) indicated.

In contrast to manual updates (Figure 8a-c), the productivity rate of the Cut Crew was updated automatically using the Bayesian inference approach previously described (Section 4.4.2.1). Task durations from as-built data in the *Actual Task Time Input Table* (Figure 8d; Figure 9, histogram) were—together with the prior predictive distribution (Figure 9, dashed)—used to generate the posterior predictive Bayesian distribution (Figure 9, solid) from which productivity rates in subsequent simulation runs were sampled.

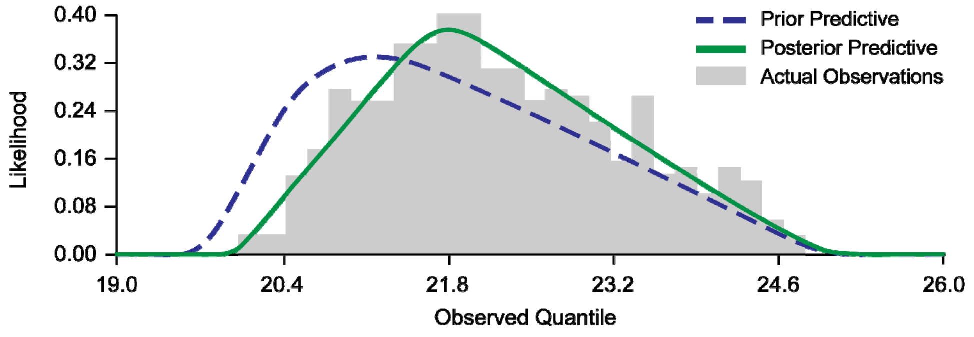


Figure 9: Actual observations (histogram) as well as the prior (dashed) and posterior (solid) predictive Bayesian distributions for Cut Crew productivity.

5.5 Verification and Validation

Preliminary verification of the proposed environment was first performed using a simple problem consisting of four products, two templates, and eight tasks. Results were calculated manually (using a spreadsheet) and were compared to those derived using the dynamic simulation environment. Outputs of the environment matched the hand-derived solution. Notably, the simple problem was converted to an automated unit test. Functionality and validity was further examined by applying the proposed environment to solve more realistic problems. Datasets adapted from real project data were used to develop two illustrative examples—an industrial pipe spool fabrication application (Section 5) and an industrial module erection model. Results were successfully obtained in both examples and were determined to be reasonable using face validation.

6 DISCUSSION

This work is attempting to address existing challenges in construction simulation through the development of a new simulation environment capable of executing dynamic, data-driven simulation without the need for routine manual intervention. The proposed environment uses advanced metadata structures to drive model self-adaptation and Bayesian inference for automated parameter updating. The purpose of this paper is to introduce the proposed environment and to describe the preliminary work that has been undertaken to transform the conceptual model into an implementable tool.

Preliminary results have demonstrated the ability of the proposed environment to generate functional models and outputs from simplified project datasets. In its present form, the proposed environment is limited to the automated updating of productivity parameters and is not capable of automatically updating model flow. Future work, which is currently ongoing, is expanding the functionality of the proposed environment to include automated updating of model structure, flow, and non-productivity associated parameters (e.g., resource availability). As the development of the simulation environment progresses from a conceptual model into an implementable tool, additional verification and validation will be performed to evaluate the accuracy, robustness, and functionality of the resulting system—including the practical application of the proposed environment to a real construction project.

ACKNOWLEDGMENTS

This research is sponsored by the Natural Sciences and Engineering Research Council of Canada (NSERC) through a Collaborative Research and Development Grant (CRDPJ 492657).

REFERENCES

AbouRizk, S. 2010. "Role of Simulation in Construction Engineering and Management." *Journal of Construction Engineering and Management* 136(10):1140-1153.

AbouRizk, S., S. Hague, R. Ekyalimpa, and S. Newstead. 2016. "Simphony: A Next Generation Simulation Modelling Environment for the Construction Domain." *Journal of Simulation* 10(3):207-215.

Ang, A. H-S., and W. H. Tang. 2006. *Probability Concepts in Engineering: Emphasis on Applications to Civil and Environmental Engineering* 2nd ed. New York, New York: John Wiley & Sons, Inc.

Chung, T. H., Y. Mohamed, and S. AbouRizk. 2006. "Bayesian Updating Application into Simulation in the North Edmonton Sanitary Trunk Tunnel Project." *Journal of Construction Engineering and Management* 123(8):882-894.

CII. 2021. Project Controls. https://www.construction-institute.org/resources/knowledgebase/knowledge-areas/project-controls, accessed April 21, 2021. Austin, Texas: Construction Industry Institute.

Hague, S., and S. AbouRizk. 2019. "Nonparametric Frequency Polygon Estimation for Modeling Input Data." In *Proceedings of the 18th International Conference on Modeling and Applied Simulation*, edited by A. G. Bruzzone, L. M. S. Dias, F. de Felice, M. Massei, and A. Solis, 159-165. Genoa, Italy: DIME University of Genoa.

Halpin, D. W. 1973. *An Investigation of the Use of Simulation Network for Modeling Construction Operations*. Ph.D. thesis, Department of Civil Engineering, University of Illinois, Urbana-Champaign, Illinois.

Halpin, D. W., and L. S. Riggs. 1992. *Planning and Analysis of Construction Operations*. New York, NY: John Wiley & Sons, Inc.

Lee S., A. Behzadan, A. Kandil, and Y. Mohamed. 2013. "Grand Challenges in Simulation for the Architecture, Engineering, Construction and Facility Management Industry." *Computing in Civil Engineering* 2013*:*773-785.

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):04016035.

Lu, M., F. Dai, and W. Chen. 2007. "Real-Time Decision Support for Planning Concrete Plant Operations Enabled by Integrating Vehicle Tracking Technology, Simulation, and Optimization Algorithms." *Canadian Journal of Civil Engineering* 34(8):912-922.

Lugaresi, G., and A. Matta. 2018. "Real-Time Simulation in Manufacturing Systems: Challenges and Research Directions." In *Proceedings of the 2018 Winter Simulation Conference*, edited by M. Rabe, A. A. Juan, N. Mustafee, A. Skoogh, S. Jain, and B. Johansson, 3319-3330. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Microsoft Corporation. 2021. Visual Studio. https://visualstudio.microsoft.com/vs/, accessed June 28, 2021. Albuquerque, New Mexico: Microsoft Corporation.

Song, L., F. Ramos, K. Arnold. 2008. "A Framework for Real-Time Simulation of Heavy Construction Operations." In *Proceedings of the 2008 Winter Simulation Conference*, edited by S. J. Mason, R. R. Hill, L. Mönch, O. Rose, T. Jefferson, and J. W. Fowler, 2387-2395. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Waschneck, B., T. Altenmüller, T. Bauernhansl, and A. Kyek. 2016. "Production Scheduling in Complex Job Shops from an Industry 4.0 Perspective: A Review and Challenges in the Semiconductor Industry." In *Proceedings of the 1st International Workshop on Science, Application, and Methods in Industry 4.0*, edited by R. Kern, G. Reiner, and O. Bluder. Graz, Austria: Know-Center.

Zhang, C., A. Hammad, and S. Rodriguez. 2012. "Crane Pose Estimation Using UWB Real-Time Location System." *Journal of Computing in Civil Engineering* 26(5):625-637.

AUTHOR BIOGRAPHIES

RAMZI ROY LABBAN was a Postdoctoral Fellow in the Department of Civil and Environmental Engineering at the University of Alberta. He is currently a Senior Advisor at SMA Consulting. His email address is labban@ualberta.ca.

STEPHEN HAGUE is a System Analyst in the Department of Civil and Environmental Engineering at the University of Alberta. His e-mail address is steve.hague@ualberta.ca.

ELYAR POURRAHIMIAN is a PhD Student at the University of Alberta studying Construction Engineering and Management. His email address is elyar@ualberta.ca.

SIMAAN ABOURIZK is a Distinguished University Professor, Interim Dean of the Faculty of Engineering at the University of Alberta, and a Tier 1 Canada Research Chair in Operations Simulation. His email address is abourizk@ualberta.ca.