Proceedings of the 2003 Winter Simulation Conference
S. Chick, P. J. Sánchez, D. Ferrin, and D. J. Morrice, eds.

MODELLING A CONTINUOUS PROCESS WITH DISCRETE SIMULATION TECHNIQUES AND ITS APPLICATION TO LNG SUPPLY CHAINS

Niels Stchedroff
Russell C.H. Cheng

Faculty of Mathematical Studies
University of Southampton
Southampton, SO17 1BJ, U.K.

ABSTRACT

This paper discusses the problem of modelling an LNG supply chain efficiently. The production, processing, transportation and consumption of LNG (Liquid National Gas) and the associated products are a topic of major interest in the energy industry. While the problem is apparently continuous, analysis suggests that this problem can be modelled using discrete, deterministic techniques. A method involving a modification of the Three Phase discrete technique was used. Analysis of the way in which the effects of an event spread leads to a method by which excessive re-calculation can be avoided, yielding a model that is computationally very efficient.

1 OVERVIEW OF THE PROBLEM

An LNG supply chain consists at the highest level of loading ports shipping LNG to one or more receiving ports. A typical supply chain is depicted in Figure 1.

The loading ports and receiving port structures are depicted in more detail in Figures 2 and 3 respectively.

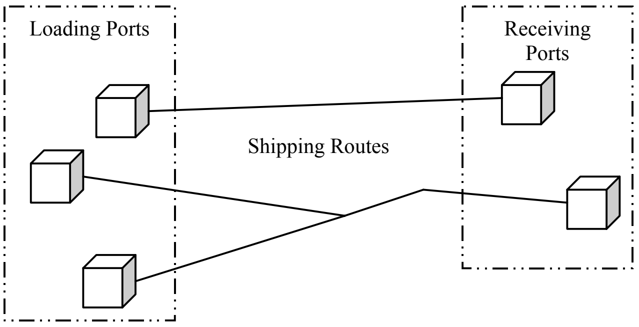

Figure 1: LNG Supply Chain Structure – High Level

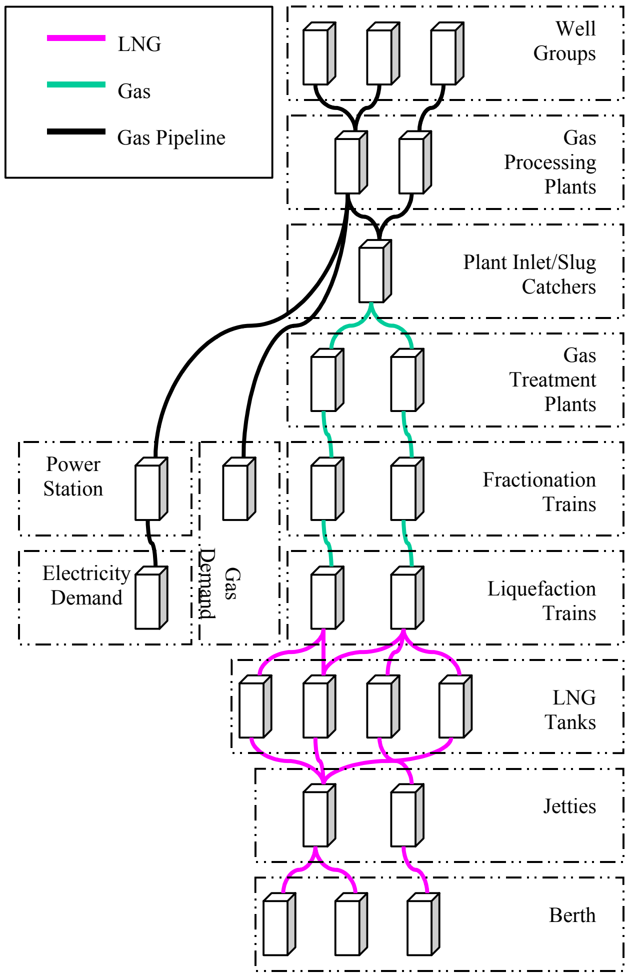

Figure 2: Loading Port Structure

2 OBJECT STRUCTURE AND STATES

In the above supply chain structure all the key objects to be modelled possess a very similar structure. The assumptions in this object representation are an extension of one of the author's work with the ADGENT series of models for Shell.

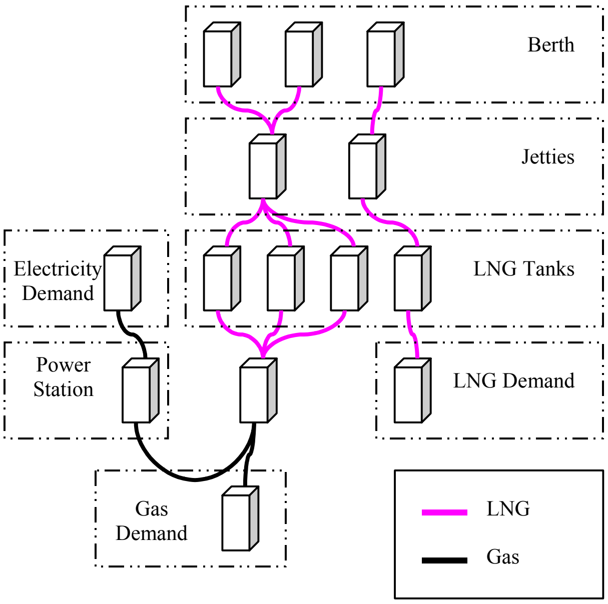

Figure 3: Receiving Port Structure

It will be seen in Figures 2 and 3 that within the port structure at both ends are a number of substructures, such as tanks, pipelines and processing equipment. Each item has the following basic properties illustrated in Figure 4, and Table 1.

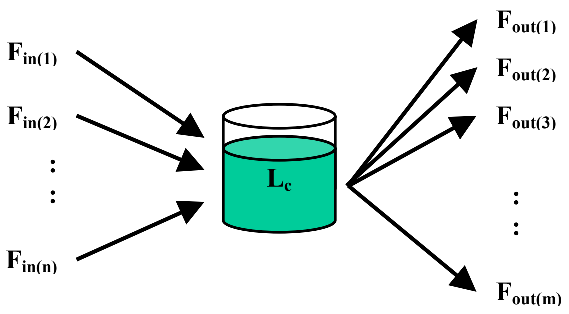

Figure 4: Basic Item

Table 1: Basic Item Properties

$F_{in(n)}$	Flow into the object from another object
$F_{out(m)}$	Flow out of the object to another object
L_c	Current level in the object.

Note that pipelines and connectors are also treated as objects under this definition. The objects that we will define have the following properties:

- Input into objects: The input value are the amounts of material flowing out of the connect objects.
- Output from objects: The output values are the amounts of material that the objects connected to can accept.
- Material level inside an object.

The level at any future point can be calculated as:

$$l_f = \left(\sum_{i=1 \text{ to } n} F_{in(i)} - \sum_{j=1 \text{ to } m} F_{out(j)} \right) \times (t_f - t_c) + l_c$$

Where t_f = the future time point, t_c = the current time point, l_c = the material level at the current time, l_f = the material level in the object at time t_f.

From the above, we can see that the following basic states that need to be considered.

2.1 Object Maximum

In this state, the object has been filled to its maximum internal capacity. As a result, the inputs must be scaled back so that

$$\sum_{i=1 \text{ to } n} F_{in(i)} \leq \sum_{j=1 \text{ to } m} F_{out(j)}$$

Dividing the normal total output by the total input that the connecting objects can provide does this. This gives the ratio of input that is possible to the ratio of input that is being supplied. For example, if the total input can only be 60% of the possible amount, then each input is scaled back to 60% of the possible amount:

$$F_{new\_in(x)} = \left(\sum_{i=1 \text{ to } m} F_{out(i)} \Big/ \sum_{j=1 \text{ to } n} F_{in(j)} \right) \times F_{in(x)}$$

2.2 Object Minimum

In this state the object has reached the minimum internal level. As a result, output must be scaled back to match input, as in the Object Maximum. This is done (as before) by calculating the ratio between the total possible and the total actual, and using it to scale the outputs accordingly:

$$F_{new\_out(x)} = \left(\sum_{i=1 \text{ to } n} F_{in(i)} \Big/ \sum_{j=1 \text{ to } m} F_{out(j)} \right) \times F_{out(x)}$$

2.3 Object in Restricted Operation

In this state the object can only pass through a proportion of its capacity. Outputs are scaled back accordingly:

$$F_{new\_out(x)} = \left(\sum_{i=1 \, to \, m} F_{out(i)} \right) \times e_{effect}$$

2.4 Object in Normal Operation

The object has space from all the input from the connecting objects, and capacity to satisfy all the outputs.

2.5 Ships

Ships can be considered as above when they are in port, loading or unloading. In the former case, there are no outputs, in the latter, no inputs. Leaving and entering ports creates an event, which disconnects the ship, causing the jetties etc to recalculate their in/out flow rates.

3 CHOICE OF SIMULATION RUN METHODOLOGY

The equations discussed in the previous section are extremely simple. The events that cause state changes occur at discrete points in time. This means that calculating the future states of the system does not require solution of differential or integral equations. In this case continuous methods are not required, leading towards selection of a method based on discrete modelling techniques. In the case of FLEET, the ADGENT models for Shell and some systems created by Lanner in WITNESS, shipping systems are modelled in this discrete manner, using changes in state at specified times to represent the various operations.

Efficiency is an important concern. When validating a configuration or using optimization techniques to create a configuration hundreds or thousands of runs may be required. Users of this kind of model have also expressed an interest in being able to manipulate the results generated by the system and using the model to validate the changes. To be usable this would require nearly instantaneous recalculation

A given model can be implemented using anyone of four basic approaches: Event, Process and Activity based approaches (Mitriani, 1982; Pidd 1998), the Three Phased approach and System Dynamics (Pidd). The approaches are all equivalent in the functional sense – a given model can be implemented in any one (Perumalla & Fujimoto, 1998; Pidd 1998). We have preferred the three-phase approach as this has several advantages.

- The dead locking problems of the process-based approach are avoided.
- The inefficiency of the activity based approach trying the test head for every activity is avoided.
- The complexity of modelling interaction in the event based method (where each event routine must contain the actions required to deal with interaction), is dealt with in the handling of the conditional events.
- System Dynamics employs time slicing and so is less efficient.

The key to the three-phase approach is to divide the way an activity starts into two event types – Conditional and Bound. Bound events occur at particular, pre-computed times. Conditional events are affected by other factors, such as the availability of resources. The efficiency of our simulation model depends very much on an effective choice of conditional events, and we discuss in the next Section.

4 EVENT TYPES

Our main focus is on the choice of C- events, but we comment first on the B events.

B events in the classic Three-Phase model are those that have their start or finish time predicted in advance (Pidd). This can apply not only to inherently deterministic events such as darkness and tides but also probabilistic events such as weather and breakdowns. Pre-computing the probability of occurrence, using a given distribution, can accomplish this.

At the start of a run, the B events will be computed and a list of them constructed, ordered on start time. In our model B events are actually pairs - a start and an end event. Thus in our case a B event is in effect two events.

The C events are those are conditional in nature – that is they occur as a result of internal operation of the model. For the LNG supply chain model they are the key to the construction of an efficient model. The list of C events is given in Table 2.

Table 2: C Event Types

C_{MaxVol}	Object reaches maximum internal level
C_{MinVol}	Object reaches minimum internal level
C_{Normal}	Object returns to a situation where it is neither at the maximum or minimum internal level
C_{sail_in}	Ship enters port.
C_{sail_out}	Ship leaves port.

Due to the continuous nature of the model, it is however, possible to compute when these events will occur (and when the event effect will end) at a given time point unless something else happens. Again, this is different to the classic Three Phase system, where C events generally have a matching B event. In this case, the matching B event has to be recomputed dynamically during the course of the run – which makes them difficult to handle as standard B events. For this reason they are considered as a part of their parent C events.

It is possible to predict the occurrence of the next C event providing a B or C does not occur first. This becomes very important when we consider the effects on other objects of events.

5 HOW THE EFFECTS OF AN EVENT SPREAD

Let us consider the case of a C_{MaxVol} event affecting a static (non-ship) object connected into a supply chain, such as in Figure 5.

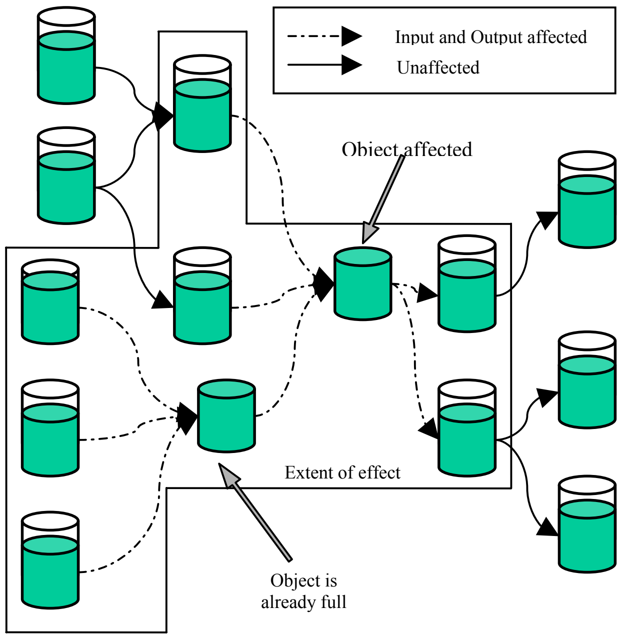

Figure 5: Objects Affected by the Event

The model reduces inflow into the object affected. This in turn affects the objects that feed the object in question. In particular it will affect the rate at which these objects are filling or emptying themselves. In other words, the time to the next C_{MaxVol} or C_{MinVol} is modified.

This does not directly affect the other items in the supply chain yet. Since we are not trying to predict the effect of on C event on the timing of another, we can consider them in isolation – what we are interested in is the time of the next event that will occur.

This also holds good for all other C events – if fact for both B and C events. Consider that in each case the events modify the operating capability of an object. When estimating the time of the next C event for each object, only the current input and output values need to be considered. This is because the estimates for the times of the C events are updated when they are affected by a change in object performance.

Note the case of a neighbouring object inputting into the object affected by the C event that is full itself. Here, the input rate into the neighbouring object is affected – in turn affecting its inputs. This chain of affects will only continue so far as the effects of the event disturb the other side of the object, so to speak.

This is fundamental to the efficiency of this approach – we only need consider the effect of an event on the neighbours of the object affected, as well as the object itself. This means that a blanket recalculation of C events is not necessary every time an event (both B and C type) executes.

After an event (of either type) occurs the C events for that object and the objects immediately affected by it are recalculated.

The executive operates in three phases – the first (A phase) finds simulation time point when the next event will occur. The second (B phase) executes all the Bound tasks that due. The third phase (C Phase) tries all the outstanding Conditional events, to see if the required conditions have been met. (Law & Kelton, 2000, Pidd 1998)

6 THE EXECUTIVE

6.1 Pre Processing

The ordered list of B events is created, and all the objects are scanned to find the first C event – there may be more than one C event occurring at this time point, of course.

6.2 A Phase

The timing of the next event is calculated. This will either be the time at which the next B event(s) in the master list will start, or the time of the next C event. The simulation clock is advanced to this point.

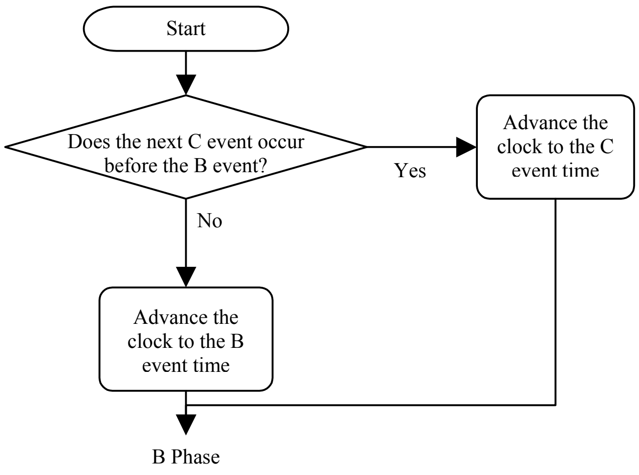
Start
Does the next C event occur before the B event?
Yes
Advance the clock to the C event time
No
Advance the clock to the B event time
B Phase

Figure 6: A Phase

6.3 B Phase

As we have seen, the Bs in fact have both a start and end time specified. In turn that means that to find the current list of active events we add the events whose start times are now greater that the clock time. We also remove events whose end time is now less than the current simulation clock time. Figure 7 shows the algorithm for this.

First we check to see if there are any B events to be executed. If there are, recalculate the current levels in the affected object, and then execute the events. Then the C events times for the object affected are recalculated – see “How the effects of an event spread”, above. If one of the recalculated C events occurs before the current next C event, substitute it. Finally, we recalculate the current levels in the affected.

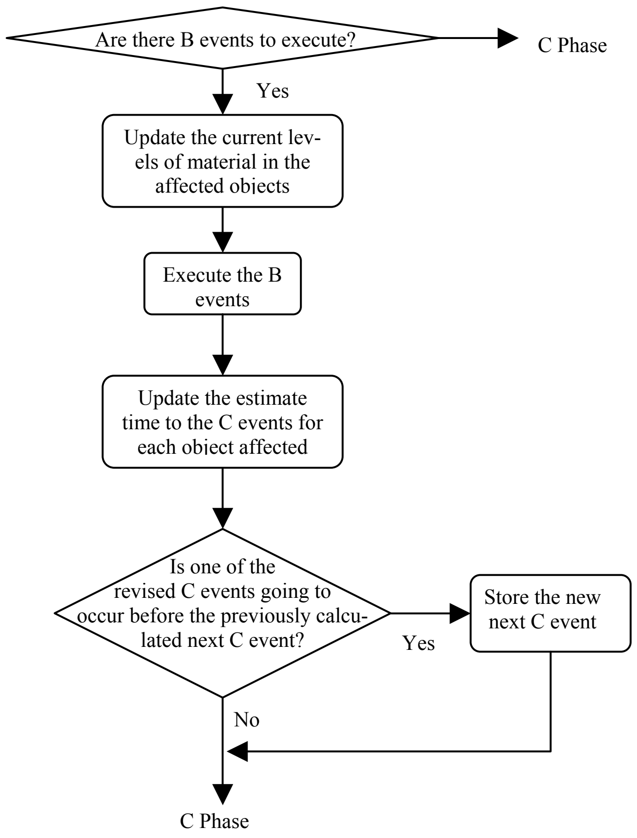
Are there B events to execute?
C Phase
Yes
Update the current levels of material in the affected objects
Execute the B events
Update the estimate time to the C events for each object affected
Is one of the revised C events going to occur before the previously calculated next C event?
Yes
Store the new next C event
No
C Phase

Figure 7: B Phase

6.4 C Phase

The next C event will have been calculated either by the pre processing (above) or by the re-calculation processes outlined below.

We have seen that the B events are essentially external to the model when it is running. The C events are those that occur as a result of the operation of the model.

By the time we reach to C phase, the performance of the various parts of the model will be up to date – flow rates, ship speeds etc. This means that we can calculate the point in time at which the next C event will occur

At the beginning of the simulation we calculate the start times of all the C events. This yields the time of the first C event. There after, the time to the next C event is revised each time an event affects an object. However, as we have seen, we do not need to recalculate the times for every C event. Instead, we recalculate the C event times for objects that have been affected by the event occurrence.

7 CONCLUSION

The method outline above allows the efficient modelling of an apparently continuous problem, using linear programming. In turn this gives the modeller a wide choice for the language/modelling environment that he chooses to use.

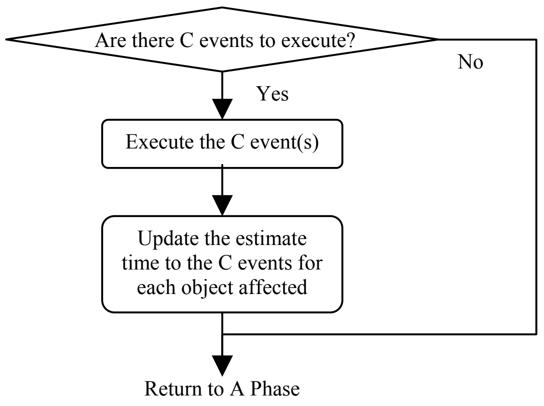
Are there C events to execute?
No
Yes
Execute the C event(s)
Update the estimate time to the C events for each object affected
Return to A Phase

Figure 8: C Phase

The objection might be advanced that the rules are overly simplistic. To my knowledge this level of complexity has been used successfully in several LNG modelling systems to my knowledge. Further, if it is desired to create a model with more detail, the basic concepts can be extended – more C events to match increased numbers of object states, for example, or more complex equations governing the rates at which objects fill.

REFERENCES

Law, A.M. & Kelton, W.D. 2000 *Simulation Modeling and Analysis*, McGraw-Hill.

Mitriani, I 1982 *Simulation Techniques for Discrete Event Systems,* Cambridge University Press.

Perumalla, K & Fujimoto, R 1998 Efficient Large-Scale Process-Oriented Parallel Simulations. In *Proceedings of The Winter Simulation Conference*, ed. D.J. Medeiros, E.F. Watson, J.S. Carson & M.S. Manivannan, 459-466

Pidd, M. 1998, *Computer Simulation in Management Science,* Wiley.

AUTHOR BIOGRAPHIES

NIELS STCHEDROFF is an IT Consultant for Shell Information Technology International. He has a BSc in Computer Science from University College London and is an MPhil student at the University of Southampton. He has worked extensively in the field of capacity and operations planning.

RUSSELL C. H. CHENG is Professor, Head of Operational Research, and Deputy Dean of the Faculty of Mathematical Studies at the University of Southampton. He has an M.A. and the Diploma in Mathematical Statistics from Cambridge University, England. He obtained his Ph.D. from Bath University. He is a former Chairman of the U.K. Simulation Society, a Fellow of the Royal Statistical Society, and Member of the Operational Research Society. His research interests include: variance reduction methods and parametric estimation methods. He was a Joint Editor of the *IMA Journal of Management Mathematics*.