Proceedings of the 2012 Winter Simulation Conference
C. Laroque, J. Himmelspach, R. Pasupathy, O. Rose, and A. M. Uhrmacher, eds

ACTIVITY BASED SCHEDULING SIMULATOR FOR PRODUCT TRANSPORT USING PIPELINE NETWORKS

Danilo Shibata
Daniel Alfenas
Ricardo Guiraldelli
Fundação para o Desenvolvimento Tecnológico da Engenharia - FDTE
R. Padre Eugênio Lopes, 361
São Paulo 05081-000, Brasil

Marcos R. Pereira-Barretto
Escola Politécnica, Universidade de São Paulo
Av. Prof. Melo Morais, 2231
São Paulo 05508-300, SP, Brasil

Fernando Marcellino
Petróleo Brasileiro S.A. – PETROBRAS
Av. Paulista, 901 – 4º andar
São Paulo, 01311-100, Brasil

ABSTRACT

Oil companies often rely on scheduling algorithms to increase the throughput of oil derivatives and other products which are transported through pipeline networks. This work presents an architecture for a scheduling simulator for pipeline networks, and outlines the rules for a method that is used in that simulation. Its core was developed as part of a decision support system that assists its users to face a very difficult challenge: how to operate a large pipeline network in order to adequately transport products from refineries to local markets. We describe the problem that led to the development of that methodology, the model and architecture of the simulator, in addition to elaborating further on the methodology which is the simulator cornerstone. Finally, a simulation example is presented as well as the results of this research.

1 INTRODUCTION

Petrobras owns the largest networks of pipelines used for oil derivatives distribution in Brazil and one of the most complex in the world. This network supplies the Brazilian states of São Paulo, Minas Gerais and Goiás, and also Distrito Federal which add up to a total population of more than sixty million people. This network is also relied on for the transportation of ethanol, one of the four most used automotive fuels in Brazil, along with Gasoline, Diesel and LPG. Supplying for such a demand of products is a complex task, that requires integrated planning of refineries, delivery places, movement of ships and the movement of products through the pipeline network. Employees responsible for scheduling the product movement must handle a large number of restrictions and various types of unexpected drawbacks, like ship delays, problems on pumping devices and changes on production planning.

This work presents an overview of the model and architecture of the simulation tool, while focusing on the algorithm used in the *Pipeline Simulator*, a discrete method based on activities for simulation of pipeline transportation which calculates the consequences of pumping sequences obtained as solution of scheduling problems. The calculated consequences consist of product movement in and out of the pipes and products that are delivered at another part of the system.

978-1-4673-4781-5/12/$31.00 ©2012 IEEE

2 PROBLEM AND OBJECTIVES

Induced by the oil market growth in Brazil, Petrobras faces the challenge of optimizing the usage of their pipeline networks in order to be able to completely supply for any consuming markets that relies on their services. The company keeps a logistics branch that handles this task with the assistance of decision support systems which allow users to choose the best planned schedules for their pipeline networks. Scheduling is accomplished collaboratively among many professionals, so that it creates an additional complexity to the problem, the unification of the changes in a way that the whole is also correct thus avoiding inconsistencies and pitfalls in the final result which could lead to eventual problems in real world operation. This leads to the necessity of a tool that can validate obtained solutions, checking for any problems and informing users in timely manner so that they can make corrections before the definitive solution is published for operation. A simulation tool that receives a schedule, simulates the transportation that would result from its implementation in the real world and informs the operator about any observed problems is then needed.

Petrobras already has a fifteen-years-old simulator that performs such tasks, but it does not fully consider particularities of operation and lacks a number of features that would increase its usability. Other works, which were presented over the last ten years, have different foci, either were the base of an optimization tool or considered a smaller set of restrictions (Amado 2011). This led to the development of a new simulator, that uses the methodology presented hereafter and satisfies the following constraints:

1. *Performance*: one of the aspects of fundamental importance. It is desired that the application can execute typical size simulations (one month long) in rather short time (less than a few seconds), so that it is fully responsive to the user.
2. *Evaluating*, not solving: movement planning and movement simulation should be completely separate tasks. Although the simulator calculates start time and end time for all, except the scheduled pumping activities, it does not decide what will happen in any situation. The calculated values are nothing, but a consequence of the schedule that has been previously defined and used as input to the simulator.
3. *Operation particularities*: previous works considered a narrow set of operation rules, which forces users to make up for the lacking features by manipulating data whenever they are needed. The simulator should consider a broader set of operations and restrictions than previous works considered, such as blending, injection, partial delivery operations, working shift changes, power consumption policies, etc.
4. *Network topology independence*: simulator implementation should be independent of simulator configuration, that is composed of information about the topology and scheduling. Decoupling implementation and configuration allows operators to reconfigure the application without the interference of software developers.
5. *Feeding a rich user interface*: simulated data should be presented to users in an easy to understand and to change manner. So it would include graphics and tables that can be changed with mouse and keyboard commands, and a view that allows frame-by-frame analysis of the movements. The final intent of the simulator is helping users to get the best possible scheduling, and these rich interfaces should allow users to detect and correct problems before the results are published to avoid problem propagation to other users.

2.1 Comparative Literature Review

Amaro (2011) presented an extensive coverage of *RPDPPS* (refined products distribution problem via pipeline systems), a research area that includes this work. It is strongly advised for anyone starting in the area to read his literature review instead of using the present work as an entry point.

That said, this review focuses only in comparing this work to other publications in the area. Most of them focus on solving and optimizing the scheduling problem, with limitations when considering the number of sources, destinations and pipelines, as well as the range of operations supported (bidirectional

flow, movements with blending operations, etc.), while this work is focused in simulation only, without limitations related to network topology and considers a very high number of restrictions, superior to every publication reviewed by Amaro (2011), so a direct comparison is not possible. Some of these works are based on simulation (Camacho et al. 1991, Mori et al. 2007) while many others use it as a supporting tool (Magatao, Arruda and Neves Jr. 2004.; Neves Jr. et al. 2007; Cafaro et al. 2010).

This work does not divide neither time nor volume in uniform amounts, like Cafaro et al. (2010). Also, there is no relation between the volume of batches and pipelines being used. An important characteristic is that there is no heuristic-based processing, as does Neves Boschetto et al. (2008); all algorithms are based on physics rules of simplifications of them (propagation and blending models) or statistical history (pumping flow ratings restrictions).

Also, though this work proposes to help Petrobras distribution problem, it is intended to work in any topology of pipeline networks. Other works also related to Petrobras (Moura et al. 2008, Neves Jr. et al. 2007) focus on the current state of the network, and require the system to be recalibrated – possibly even its source-code to be changed – in case any changes in the topology occurs.

3 ARCHITECTURE

The proposed simulator consists of a software application capable of simulating product transportation, controlling amounts of products stored on depots and tanks and generating errors or warnings whenever transportation or storage does not comply with defined constraints. It is a multi-module application consisting of *Pipeline Simulator*, *Depot Simulator* and *Storage Simulator*. Each of the modules is supplemented by a specific validator that checks whether or not the simulation is correct and generates warnings when constraints are not satisfied.

The *Pipeline Simulator* is the core module of the system. This module receives as input the initial blocks of products stored in the pipes, the pumping sequence to be executed and the strategies of movement for each batch. The size of these batches are not restricted: any volume, even a fraction, is a valid batch size value, a remarkable difference if compared to other works (Cafaro et al. 2010). The strategies provide information on the movement route, blending and deliveries in an abstract and uniform way, creating a model that is broader than those based only in routes (Mori et al. 2007). The simulation creates a set of activities that represent product movement as depicted in the bottom of Figure **1**.

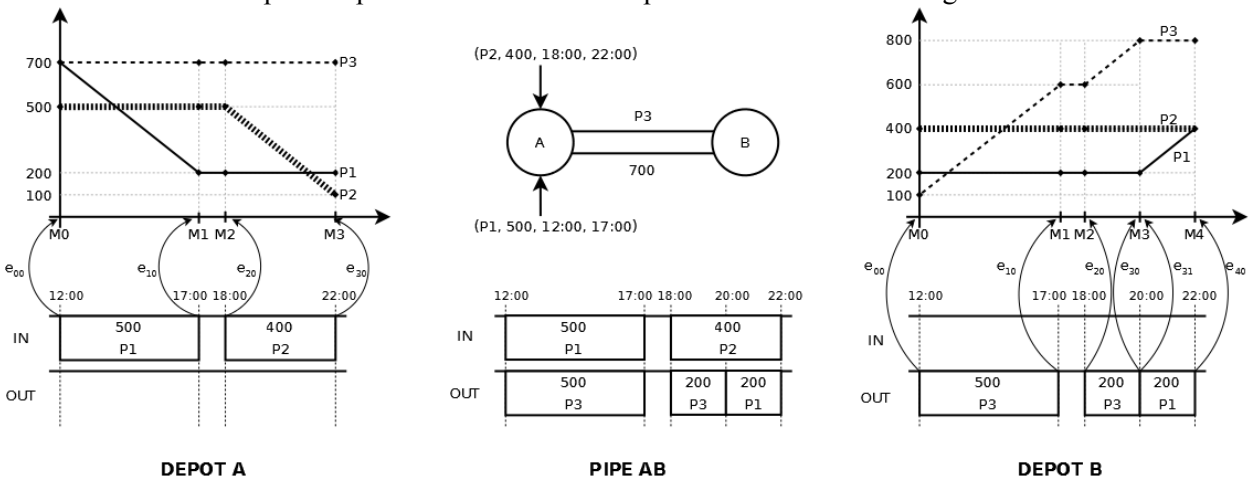

Figure 1 - Pipeline Network Transport

The *Depot Simulator* handles the activities, creates a set of events (represented by e) that represent instantaneous changes in the behavior of product storages, and groups them into milestones (M) according with the instant in which they happen. Based on the initial content of each product, the *Storage Simulator* uses the milestones and calculates the evolution of the storage as presented in the charts beside the depots.

The computation is based on the fact that between two milestones there are no changes in behavior of a storage, thus it allows to extrapolate the content at any given time using simple mathematic operations. The calculation can either be made by storage of products in tanks or in depots. More complex cases allow pipeline inputs to be merged and pipeline outputs to be divided into input and delivery activities as will be seen later in this paper.

4 MODEL

This section presents an overview of the data model, while focusing on parts used by the *Pipeline Simulator*. The following description handles movements that happen in the pipeline, pipeline content and how changes of that content are processed by the simulator, and the set of activities that represent product movement that the simulator creates and handles during its execution.

An important assumption that guided the development of this method is that all products transported in the pipes are incompressible liquids, a simplification used by other works in the area such as Milidiú, Pessoa and Laber (2002) and Lopes et al. (2007). As a consequence of this assumption, the volume of products in a pipe is constant over time and the entrance of a volume of liquid at an end of the pipe causes the same amount of volume to leave at the other end.

4.1 Movements

From the simulation standpoint, information about products stored in pipes is not enough to completely track what happens to them, so that a broader abstraction is needed to indicate how products move inside the network and when they leave it.

Movements represent batches of products that were moved inside the pipeline network, in accordance with pre-defined movement strategies. The strategy holds information about the route to be traversed including the pipes and flow reversion information, delivery rules that indicate when and where the batch should be delivered and blending configuration that indicate where blending and injection operations are expected to be executed. Movements may optionally have a parent movement to indicate that the child is a composing part of the parent movement, as it happens in blending operations when two different movements have a common parent indicating the result of those movements being blended together. This is depicted in Figure **2**, where M1 (grey) and M2 (dashed) have a common parent movement M3 (dashed grey) which is the result of blending them together.

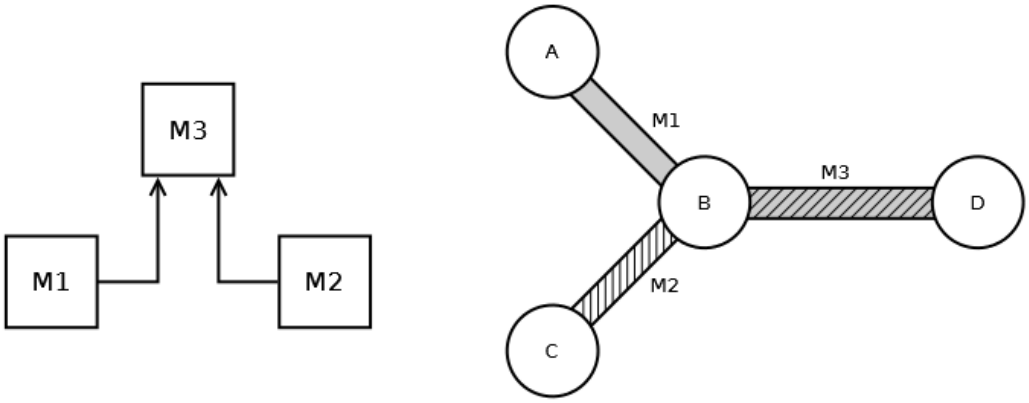

Figure 2 – Movements hierarchy due to blending operation.

Contrary to what happens in real world operation that usually keeps track of products inside the pipes and moving in and out of them, the simulator tracks pipe content and activities with references to movements. This simplifies the simulation process, since it is not necessary to track down which route the products stored in the pipes should take and which special operations apply to these products because

these information are already stored in the movement's strategy. The easiness to track product movements also simplifies understanding the simulation, as well as the fact that batches of the same movement located adjacently inside a pipe will be easily distinguishable.

4.2 Pipeline Content

Pipe content is modeled as a sequence of items that represent volumes of the batches stored in that pipe. Every time two storage items, with same product and movement, have direct contact inside a pipe, they are merged into one. Changes to pipe content are handled as if pipes were double ended queues of items, i.e., when a volume is inserted into an end of the queue, the same volume must be removed from the other end.

4.3 Activities

As stated by Muzy and Hill (2011), activities are operations that change the state of a system over time. Simulator activities can be related to product movement, maintenance of parts of the network such as tanks and pipes, power savings periods, etc. The most important activities for pipeline simulation are the ones related to product movement in the system, that the simulator processes and creates during simulation.

Movement activities are defined by the following attributes: the resource where the movement happens, the movement of the product being moved, the volume of product being moved, the start time and the end time of the activity. They are divided in two groups based on the type of resource where they happen, which may either be a pipe or a depot. Movement activities that happen on pipes also have an attribute that defines the flow orientation inside the pipe.

The four movement activities used by the pipeline simulator are described below:

1. Pipe Input: pipe input activities represent a volume of product that is pushed into a pipe as consequence of the fact that it was pumped into the system or out of another pipe.
2. Pipe Output: pipe output activities represent a volume of product that is pushed out of a pipe as consequence of the fact that some product was pushed into the other end of the same pipe.
3. Pumping: pumping activities represent a volume of product that pumped into the pipeline network from a depot. The product is taken from the tanks of that depot and pushed into a pipe.
4. Delivery: delivery activities represent a volume of product that is pushed out of the pipeline network and into a depot. The product is pulled out of a pipe and stored into tanks at the receiving depot.

5 PIPELINE SIMULATION

As stated before, the *Pipeline Simulator* calculates the movement activities caused by a pumping sequence in a pipeline network. This section presents the pipeline simulation method, indicating how activities are calculated and later processed.

5.1 Simulation Process

Pipeline movement simulation is based on an iterative method that analyses activities and calculates their consequences (which are also activities), repeating this process until all activities have been processed or a user defined horizon has been reached. The activities are processed in ascending order based on their start time attribute, and activities that start beyond the horizon are ignored.

Processing is depicted on the `PROCESS` pseudo-code presented below. It defines two arguments, a priority queue which stores the activities to be processed ordered by their start date and a horizon date that limits the activities to be processed. Activities are taken from the queue and processed until the queue is empty. Processing of the activities will calculate new activities which will be stored in the queue as well, in order to be processed when they are due.

```
PROCESS(queue, horizon)
DO
   activity <- DEQUEUE[queue]
   process(activity, queue)
WHILE (queue NOT EMPTY AND horizon > START[activity])
```

The queue should be initialized with all pre-scheduled pumping activities into it, while the horizon should be set to a date after the initial date of the scenario. The pseudo-code assumes that when processing starts the queue is not empty since it does not make sense to simulate if no activities exist, and that the horizon is always defined since there will always be a large enough date that allows for processing all activities (any date after the end of the pumping activity that ends last).

5.1.1 Pumping

Product transportation through pipeline networks begins when a product is pumped from a depot into the network, which means the product is taken from a tank in the depot and pushed into one of the pipes of the network that is connected to that depot. Although all movements that happen inside the pipeline are consequences of products that are pumped into pipes elsewhere in the network, the direct consequence of a pumping activity (P) is an amount of the product that is pushed into a pipe.

The processing of pumping activities is depicted by Figure **3**, a pipe input activity (PI) is created by copying the movement, volume, start time and end time of the pumping activity, while traversed pipe and orientation are retrieved from the strategy of the movement. The new pipe in activity is stored in the queue for processing further in the simulation.

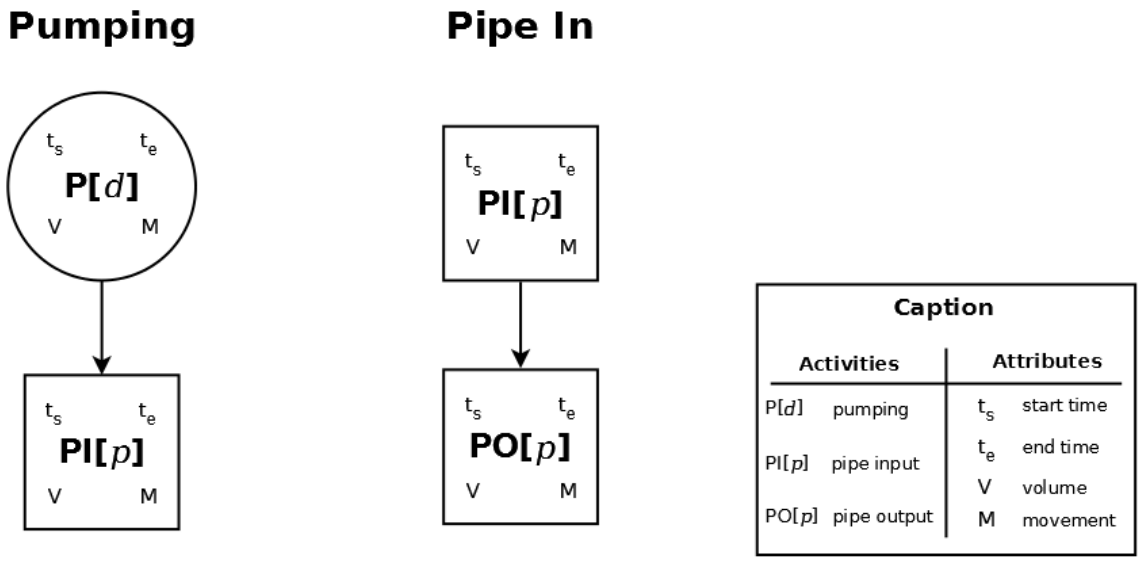

Figure 3 - Processing of Pumping and Pipe In Activities.

5.1.2 Pipe Input

Product movement inside the pipeline network comprises of products that are pushed in and out of the pipes. Products are pushed into pipes as consequences of the fact that they are pumped into the network from a depot, or pushed out of other pipes. The consequence of a product being pushed into a pipe is the

same volume of fluid that is pushed out in the other end of the same pipe. If two products are pushed into the same pipe at the same time, then a blending operation occurs. If one of the blended parts is an immediate consequence of a pumping activity, then it is an injection operation.

The processing of pipe input activities that do not demand blending operations to be performed is depicted in the Figure **3**. Similar to what happens in the processing of pumping activities, the attributes of the activity are copied to a new pipe output activity, but in addition traversed pipe and orientation are included. The new activity is also stored in the queue to be processed later.

Blending operations can be simulated by using the appropriate movement strategies, which define a configuration that indicates where the blending is expected to happen. The result of a blending operation is defined as the first common ancestor of the two movements that are blended, and the lack of a common ancestor indicates an error.

Once a conflict is detected, the processing attempts to divide the conflicting activity in three parts as depicted in Figure **4**: an optional part prior to the conflict (CUT) that has been processed and should not change; a required part where blending happens (BL); and an optional part after the blending (LO) with the leftover of one of the blended parts which is rescheduled for processing. Since the part prior to the conflict will not be reprocessed, while others will, the consequences of the conflict are also cut and kept as consequences of this part if it exists, or discarded otherwise. The blending part is then reprocessed exactly as depicted in Figure **3**.

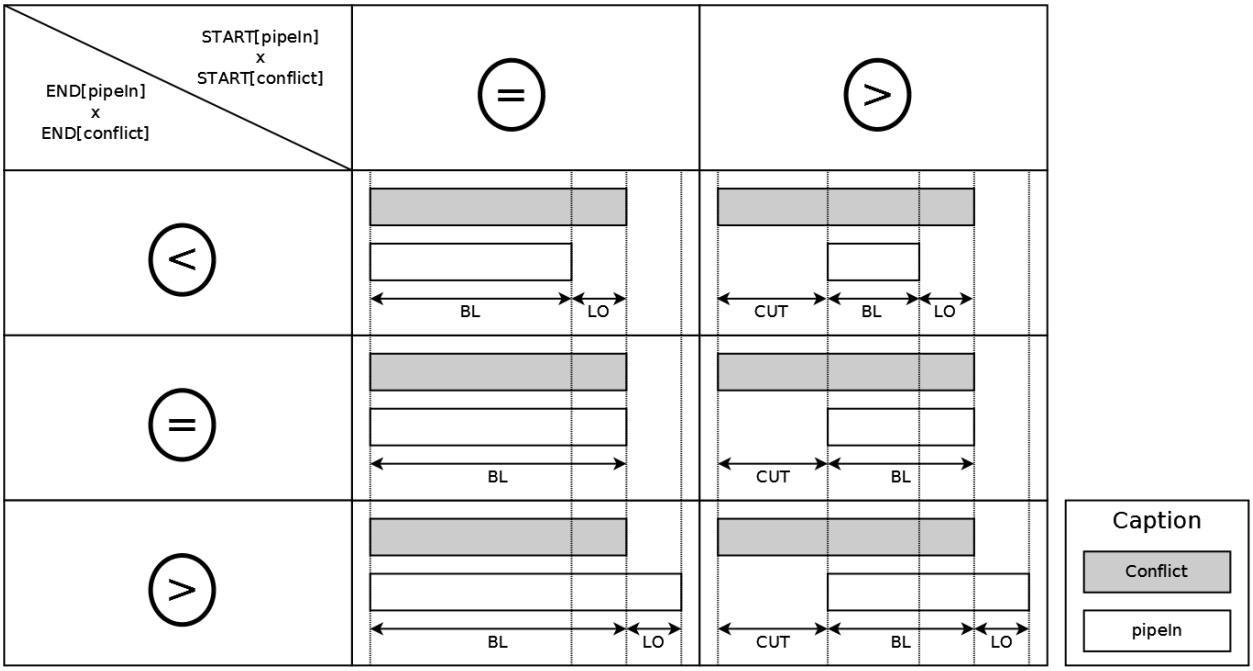

Figure 4 - Blending Operation

Injection operations are processed exactly in the same way as blending operations are, as presented above. Injections can be either synchronous or asynchronous. In the former case the pumping activities are calculated by the simulator, while in the latter case they are defined by the software users.

Synchronous injections define anchor values for the start and end time of the injection which indicate the ratio of volume over the total volume of a movement that must have traversed the pipe when the injection is supposed to start and end respectively. It also defines a volume ratio which indicates the injection flow ratio over the volume that traverses the pipe during the injection. Synchronous injections calculation uses these ratios to calculate start and end times for the injection and the volume that is pumped in accordance with the volume that is traversing the pipe, fits the pipe in activity to the calculated times by cut-

ting it, and then creates a pumping activity that matches exactly the interval in which the pipe in activity happens.

5.1.3 Pipe Output

When a product is pushed out of a pipeline, it can either be pushed into another pipe in the network, or it can be delivered. As depicted in Figure **5**, the processing is divided in two parts: a splitting part in which the activity is cut in two smaller parts if necessary, and a processing part in which the product is pushed into another pipe or delivered to a depot.

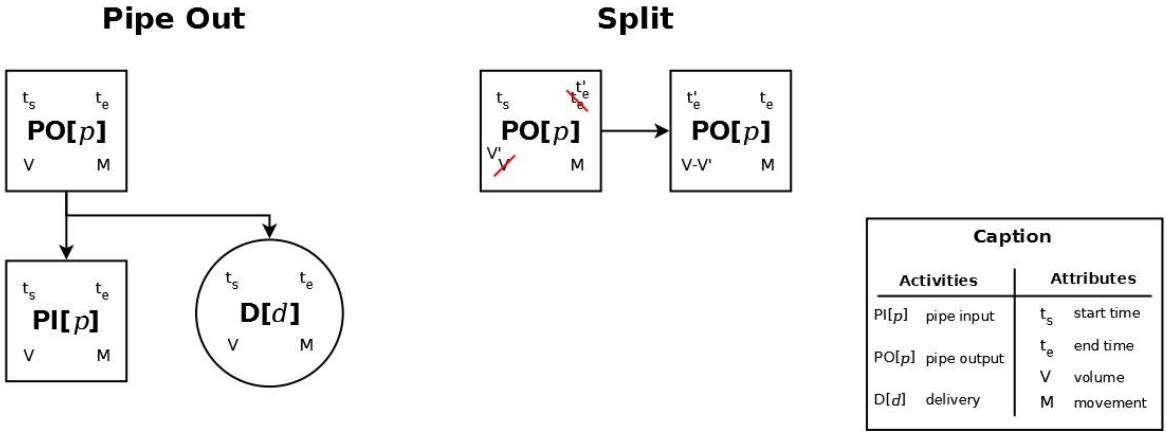

Figure 5 - Processing of Pipe Out Activity.

Pipe output activities are created as copies of the pipe input activities that generated them with the same movement, volume, start time and end time. Since the content pushed out in the other end of the pipe is not the same as the one that entered, when the pipe output activity is processed it must be fixed to reflect the content that is pushed out of the pipe. The split operation, presented in Figure **5**, checks the content at the other end of the pipe and splits the activity in two parts if necessary: the first one that reflects the last movement stored inside the pipe that is pushed out and the second that reflects the rest of this pipe out activity which will be scheduled for reprocessing later (when the process repeats itself).

Once the split operation is over, the first part is processed in order to create pipe input or delivery activities. The activities created depend on whether there is a delivery order for the movement on the depot the product is moving towards, and which type of order it is (partial or final). If there is no delivery order, the whole volume is transformed to a pipe input activity. On the other hand, if a final delivery order is defined, the whole volume is delivered at the depot. Finally, if a partial delivery order is defined, both pipe input and deliveries happen and the sum of volumes add up to the volume of the pipe out activity.

5.2 Simulation Validation

As stated before, each of the modules of the simulator has a supplementary method which validates the results obtained by the execution of that given module. In the case of the pipeline simulation, validated objects are the pipeline input activities generated by the pipeline simulation module, and maintenance periods defined as part of the simulation input.

Pipeline output activities are not validated, because they are direct consequences of pipeline input activities and the validation of such activities would generate the same results as the validation of their causes. Pumping and delivery activities are not validated, because the pipeline simulator considers that product storages are infinite, and the validation of changes, which they cause on depots and tanks, is performed later on by other validation modules.

Pipeline input activities are validated for incompatibility of products inside pipelines and transportation throughput above the maximum or below the minimum throughput defined for a product inside a given pipeline.

Maintenance periods are validated to check if the content of the pipeline during its maintenance is a maintenance product, or to check if transportation throughput does not exceed a reduced maximum throughput.

6 EXAMPLE

This section presents an example of oil derivative transportation using a pipeline network, which considers special operations of injection and partial delivery. Since injection is a particular case of blending operations, the example is helpful for understanding this type of special operation as well. The example describes the simulated scenario and presents the set of activities calculated as consequence of the simulation.

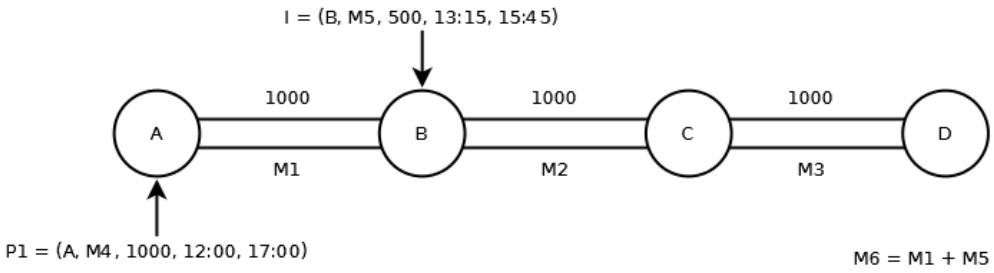

Figure 6 - Scenario for Simulation Example

The pipeline network comprises of four depots (A, B, C and D) and three pipes (AB, BC, and CD) that connect the depots as depicted in Figure **6**. The capacities of the pipes are 1000 m^3 and they are filled with fluids of movements M1, M2 and M3 respectively, which are projected to move to depot D, where they will eventually be delivered. An asynchronous partial delivery is projected to take 100 m^3 of movement M2 from the network from 1:00 PM to 2:00 PM. Movement in the network is triggered by 1000 m^3 of movement M4 that is pumped from depot A into pipe AB from noon to 5:00 PM. An injection of movement M5 over movement M1 (represented by 'I' in Figure **6**) is pumped from depot B into pipe BC from 1:15 PM to 3:45 PM.

Figure **7** presents a diagram of activities when the simulation of the scenario is over. Each of the activities presents a tag in the center that indicates the type of activity and the resource where it occurs, start and end times at the top-left and top-right corners, volume in the bottom-left corner and movement in the bottom-right corner. Activities on depots are represented by circles (tagged 'P' for pumping and 'D' for deliveries), while activities inside pipes are represented by squares (tagged 'PI for pipe input and 'PO' for pipe output). Downward arrows indicate cause/consequence relationships, while rightward arrows represent sibling activities (when two activities have a common cause). The consequences of partial delivery and injection operations are surrounded by dashed rectangles.

The movement happens from depot A to D since all batches move in that way. Deliveries happen at depots C (because of the partial delivery) and D (because of final deliveries). Movement blending happens at pipe BC due to the injection operation. Adding up the amounts delivered at depots C (100 m^3 of M2) and D (1000 m^3 of M3 and 400 m^3 of M2) will result in the same amount of product that is pumped into the network.

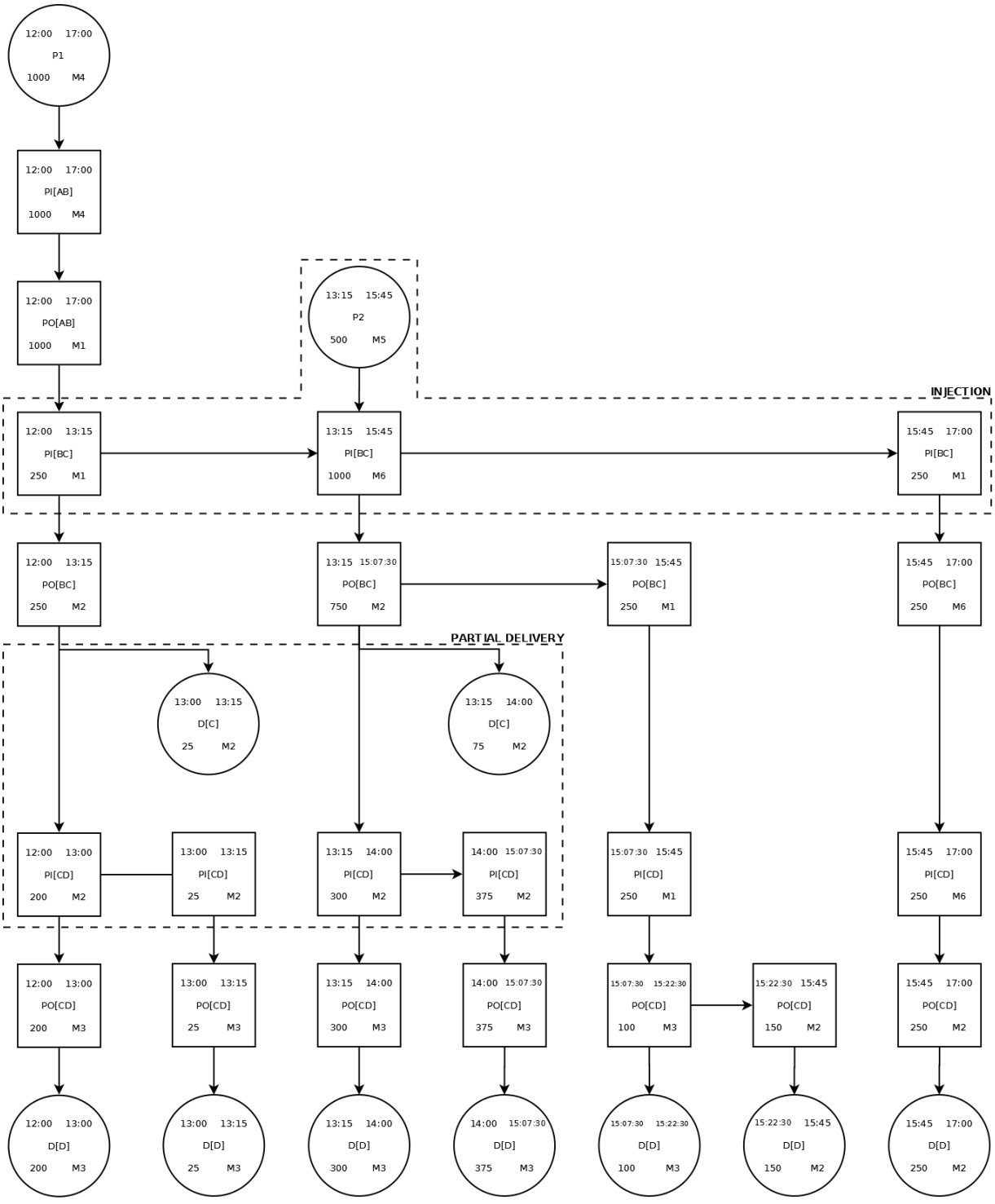

Figure 7 - Activity diagram for a simulation.

7 CONCLUSION

This work contains an activity based methodology for simulating fluid transportation using pipeline networks that was developed in a collaborative work with Petrobras as part of a software for simulation of oil derivatives distribution, which is used in a decision support system. It presented a brief overview of the

model and architecture of the simulation software, while focusing on the movement simulation algorithm which serves as the core of the application.

The simulator has been implemented and tested for multiple scenarios, including small test cases for feature testing and a real world scenario based on Petrobras' OSBRA pipeline network for a month long period. The most relevant results from the tests so far are: (1) that the simulation of the real world scenario calculates the expected activities with decent precision rounding times to the minute and (2) takes on average about 500ms in a simple laptop, a result that can be vastly improved on more resourceful servers.

The development process has been quite successful by being able to consider a broad set of operation rules used by the company, and the results obtained so far are quite useful to understand the movement process inside the pipeline network. However, the work is far from over. Future work should handle other modules of the application, the depot and storage simulators and their respective validators. Besides, eventually the rich user interface and integration to other parts of the decision support system will make this simulator fully functional.

ACKNOWLEDGEMENT

The authors gratefully acknowledge PETROBRAS for authorizing the publication of the information here presented. Furthermore, the opinions and concepts are the sole responsibility of the authors.

REFERENCES

Amado, R. J. A. 2011. "A Multi-Commodity Network Flow Approach for Sequencing Refined Products in Pipeline Systems." Ph.D. thesis, University of Tennessee, 2011. http://trace.tennessee.edu/utk_graddiss/940.

Cafaro, V. G., Cafaro, D. C, Méndez, C. A., Cerdá, J. 2010. "Oil-Derivatives Pipeline Logistics Using Discrete-Event Simulation." In *Proceedings of the 2010 Winter Simulation Conference*, Edited by B. Johansson, S. Jain, J. Montoya-Torres, J. Hugan, and E. Yücesan, 2101-2113. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Camacho, E. F., Ridao, M. A., Ternero, J. A., Rodriguez, J. M. 1991. "Optimal Operation of Pipeline Transportation Systems". In *Proceedings of the 11th Triennial World Congress of the International Federation.* 455-461.

Lopes, T. M. T., Ciré, A. A., Souza, C. C., Moura, A. V. 2007. "A Hybrid Model for Multiproduct Pipeline Planning and Scheduling Problem." *Constraints* 15:151-189.

Magatão, L., Arruda, L. V. R., Neves Jr., Flávio. 2004. "A Mixed Integer Programming Approach for Scheduling Commodities in a Pipeline". Computers and Chemical Engineering 28. 171-185.

Milidiú, R., Pessoa, A. A., Laber, E. S. 2002. "Pipeline Transportation of Petroleum Products with No Due Dates" In *Proceedings of the 5th Latin American Symposium on Theoretical Informatics*, Springer-Verlag, 248-262. London, UK.

Mori, F. M., Lüders, R., Arruda, L. V. R., Yamamoto, L., Bonacin, M. V., Polli, H. L., Aires, M. C., Bernardo, L. F. J. 2007. "Simulating the Operational Scheduling of a Real-World Pipeline Network" In *Proceedings of 17th European Symposium in Computer Aided Process Engineering*, Edited by V. Plesu and P. S. Agachi, 691-696. Amsterdam, Netherlands.

Moura, A. V., de Souza, C. C., Ciré, A. A., Lopes, T. M. T.. 2008. "Heuristics and Constraint Programing Hybridizations for a Real Pipeline Planning and Scheduling Problem", In *Proceedings of the 11th IEEE International Conference on Computational Science and Engineering*.

Muzy, A., Hill, D. R. C. 2011. "What is New with the Activity World View in Modeling and Simulation? Using Activity as a Unifying Guide for Modeling and Simulation" In *Proceedings of the 2011 Winter Simulation Conference*, Edited by S. Jain, R.R. Creasey, J. Himmelspach, K.P. White, and M. Fu. 2887-2899. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Neves Boschetto, S., Felizari, L. C., Yamamoto, L., Magatao, L., Stebel, S. L., Neves-Jr., F., Ramos de

Arruda, L. V., Lüders, R., Ribas, P. C., Bernardo, L. F. J. 2008. "An Integrated Framework for Operational Scheduling of a Real-World Pipeline Network". In *Proceedings of 18th European Symposium on Computer Aided Process Engineering*. Edited by B. Braunschweig and X. Joulia.

Neves-Jr., F., Magatão, L., Stebel, S. L., Neves Boschetto, S.,Felizari, L. C., Czaikowski, D. I., Rocha, R., Ribas, P. C. 2007. "An Efficient Approach to the Operational Scheduling of a Real-World Pipeline Network". In *Proceedings of the 17th European Symposium on Computer Aided Process Engineering*. Edited by V. Plesu and P. S. Agachi, 691-696. Amsterdam, Netherlands.

AUTHOR BIOGRAPHIES

DANILO PICAGLI SHIBATA studied computer engineering at the University of São Paulo, Brazil from 2000 to 2004. Completed his masters in computer engineering in 2008 with the studies of the use of adaptive automata for grapheme-phoneme translation for Portuguese language. Previously worked with critical systems safety analysis and software development. Currently works at FDTE with software development at project CONSUELO, developing a simulator of oil derivatives distribution through pipeline networks. His email address is d.shibata@fdte.org.br.

DANIEL ASSIS ALFENAS studied computer engineering at the University of São Paulo, Brazil, from 2000 to 2004. He has worked in all phases of software life cycle over the last eleven years. Since July 2010 he leads the engineering of sophisticated software systems, that involves simulation, optimization and rich user interface development. Currently he studies sociable robots and adaptive technologies as part of his master in computer engineering. His e-mail address is d.alfenas@fdte.org.br.

FERNANDO J. M. MARCELLINO is project coordinator in oil industry, where he has worked in modeling and development of the planning and scheduling systems of such areas as the refinery production, the distribution of oil products through pipeline networks, as well as the off-shore extraction of oil. He received his Masters degree in Computation Engineering at Technological Institute of Aeronautics (ITA), Brazil, in 2006 with the work "The Planning of the Oil Derivatives Transportation by Pipelines as a Distributed Constraint Optimization Problem". He is currently Ph.D. student at the University of São Paulo, Brazil, on the "Oil Industry Supply Chain Integrated Management through a Optimized Distributed Model". His email address is fmarcellino@petrobras.com.br.

MARCOS RIBEIRO PEREIRA BARRETTO graduated in Electronic Engineering in 1983. He achieved his M.Sc. degree in 1988 and his Ph.D. in 1993, both from the University of São Paulo. He is a professor and researcher in the Mechatronics Department at the University of São Paulo. His research includes simulation for industry applications and affective computing. His mail addres is marcos.barretto@poli.usp.br.

RICARDO HENRIQUE GUIRALDELLI studied computer engineering at the University of São Paulo, Brazil from 2004 to 2008. Completed his masters in computer engineering in 2012 with the studies of graphs applied to bioinformatics. Previously worked with development of safety critical software. Currently works at FDTE with software development at project CONSUELO, developing a simulator of oil derivatives distribution through pipeline networks. His email address is r.guiraldelli@fdte.org.br.