UML Profile for Mining Process: Supporting Modeling and Simulation based on Metamodels of Activity Diagram

Andrea Giubergia[1], Daniel Riesco[2], Verónica Gil-Costa[1] and Marcela Printista[2]

[1] Mine Department, Chacabuco 569, National University of San Luis, Argentina.
[2]Informatic Department, Ejército de los Andes 950, National University of San Luis, Argentina.

Correspondence should be addressed to: A. Giubergia; aagiuber@unsl.edu.ar

Abstract

An UML profile describes lightweight extension mechanism to the UML by defining custom stereotypes, tagged values, and constraints. They are used to adapt UML metamodel to different platforms and domains. In this paper we present an UML profile for models supporting event driving simulation. In particular, we use the Arena simulation tool and we focus on the mining process domain. Profiles provide an easy way to obtain well-defined specifications, regulated by the OMG (Object Management Group). They can be use as a pre-simulation technique to obtaining solid models for the mining industry. In this work we present a new profile to extend the UML metamodel, in particular we focus on the activity diagram. This extended model is applied to an industry problem involving loading and transportation of minerals in the field of mining process.

Keywords: Modeling, Simulation, UML Profile, Mining Process.

1 Introduction

The literature on formal use of UML (Unified Modeling Language) [OMG03], is huge and growing rapidly. This tool becomes a modeling standard in 1990 to follow what changed the world of software [W08]. UML makes it possible to model any system from different perspectives. Modeling a system from different perspectives allows to clearly picking the vision of what you want to do. This is because UML is a language that holds its own specific rules, semantic and syntax.

UML can be used as documentation of code, but it is also intended as a means of specifying a system. Model Driven Architecture (MDA) [MM03] calls for complete systems to be generated automatically from UML models. If the language is not precisely defined, the generated system may not be what the model creators intended.

UML has an abstract syntax which is processed by model transformation and code generation. The relation in UML between concrete diagrammatic syntax and the abstract syntax it represents, is complicated enough to be a potential source of error. Precisely defining this relationship could simplify the creation of graphical model editors, and facilitate animations [EHHS00] and reverse engineering. The definition should clearly delineate concrete syntax, abstract syntax and semantics, and it should also specify the relationships between these parts.

UML defines several types of diagrams to view the dynamic aspects of a system. One of these diagram types is the activity diagram [OMG07] which is used in this work to document the workflow in a system. UML Diagrams are suitable for system analysis, design and development. However, UML and its diagrams cannot accurately specify concepts for particular domains. That is why UML embodies the concept of profiles. Profiles are defined by the OMG (Object Management Group) in its specification [OMG03], as a predefined set of stereotypes, tagged values, constraints (by using OCL - Object Constraint Language), and notation icons that collectively specialize and tailor UML to a specific domain or process. When these elements are introduced the model can be clearly visualized, software developers improve communication and establish a common vocabulary. Also profiles allow add information to the model to transform it to other models. OCL (Object Constraint Language) constraints are semantic restrictions added to UML elements. The advantage of profiles is that most UML tools can easily apply them. When using profiles it is not necessary to define a special notation neither special tools (UML tool is used).

The UML metamodel (which consists of entities and relationships) of the domain is a process-oriented tool. It is graphically structured for the construction of diagrams or flowcharts. These diagrams show the number of steps required by an entity as it moves into the system.

Another powerful tool for system analysis and design is simulation. Its benefits are applicable to almost all kind of industry. It involves designing a model of a particular system, to solve it by means of a numerical technique (algorithm) and the subsequent execution of a series of experiments with the aim of understanding the behavior of such system under certain conditions [BCN96]. The model should be able to reproduce the actual process behavior as accurately as possible. There are many simulation software of general purpose [S12] [FSL13] [E10] [KSS08], in this work we focus on the Arena [KSS08] simulation because it has become the market-leading discrete-event simulation software.

To reconcile the two ways of approaching the same subject, namely UML and simulation, in this work we propose to model a system of loading and transportation of material in the field of the mining industry. In particular, we propose to extend the UML metamodel using profiles. The objective is to adapt the UML metamodel to be able to represent the basic modules of Arena.

The advantage of the proposed profile is that: a) it allows re-using the stereotypes created for the Arena simulation software. Moreover, those stereotypes can be applied to any other process-oriented simulation software. We can also use them or integrate them as part of any other process oriented simulation software.

This paper is organized as follows. Section 2 presents related work. The Arena simulation software is briefly described in Section 3. Section 4 presents our mining process case of study. Section 5 presents the proposed profile implementing a simulation environment and its OCL constraints. Conclusions are included in Section 6.

2 Related Work

Modeling and simulation techniques provide the possibility of studying new strategies and to predict the effect of new policies, new designs, and new strategies which would otherwise be too expensive or even impossible to implement and test on real cases. Both, the UML modeling and the simulation approach model from two different communities (or point of view) but they can be used for similar purposes.

Some recent methodologies and techniques for modeling and simulation have been presented in [NST09, YDHZ12, YLD13]. In particular, the authors in [NST09] use genetic algorithms to improve component analysis. This technique is applied to simulated data collected from the Tennessee Eastman chemical plant. The work in [YDHZ12] evaluates basic data-driven methods for process monitoring and fault diagnosis. Also in this case the benchmark of Tennessee Eastman (TE) process is utilized to illustrate the efficiencies of the discussed methods. Finally, in [YLD13] the authors present a fault tolerant control architecture which can be reconfigured online. The proposed scheme is evaluated with the TE benchmark model.

In this work we focus on UML modeling and simulation techniques which are widely used in systems engineering. Although these two methodologies have evolved separately, there are some works [TKMG08, BS01, BD05, JG07] that integrate modeling tools with specific simulation software.

The works presented in [TKMG08] and [BS01], show how the UML-Arena combination is used to create a model. Then this model is re-used into the simulation environment of the Arena software. In [TKMG08] the authors describe the use of activity diagrams to automatically generate simulations models and

proposes the use of an algorithm as a solution to automatically transform UML models (only activity diagrams) into simulation models. Similarly, the authors in [BS01] show the use of UML activity diagrams as the basis to build simulation models, which are then run on the Arena simulation software.

In other words, these studies show a methodology to transform activity diagrams into simulation models for Arena. However, they do not take advantage of the UML profiles. A profile created for a specific situation can be re-used into other environments with similar characteristics, thus saving time and increasing the versatility of the profile created.

As explained in [FV04], OMG defines two possible approaches for defining domain specific languages. In the first one, a new alternative language is defined [CWM]. A new tailor-made language will produce suitable specific notation that will match the concepts of a specific application domain. However, as the new language does not respect UML semantics, it will not allow the use of commercial UML tools for drawing diagrams, generating code, etc. [FV04]. The second approach (named Profiles) uses the UML metamodel respecting the original semantic of the UML elements (classes, attributes, etc.). But new constrains are added to the definition and to the relationships of those elements.

The works in [DGRM06, DMZRM12, DBRM08] show the versatility of profiles to define different environments (application domains), which extend the UML specification (not only the syntax but also the semantic through formal OCL constraints). In particular, the work in [DGRM06] shows the way for defining design patterns with profile, proposing architecture in levels. It shows how the definition of a profile for a particular pattern is, and how an UML tool can be enough for introduce profile for patterns. It analyzes the advantages of using profiles to define, document, and visualize design patterns.

The work in [DBRM08] shows the definition of the C&K metrics applied to Aspect Oriented Design (AOD) using UML profiles. A new definition of metrics using profiles without modify the OMG metamodels is specified. An OCL formal specification is developed. The defined profile allows applying methodologies which use AOD and allows measuring the design. The calculation of the resources is an activity that can improve the software development. The engineers can import this profile to any UML tool and measure the quality of its AOD. They do not need build a specific tool for AOD. They can use any UML tools and thereby improve the quality of aspect oriented developments.

In [DMZRM12], the creation of components of the Java EE 6 business platform from technical business processes modeled with Business Process Model Notation (BPMN) 2.0 is proposed. This generation was achieved by performing three transformations in the context of Directed Model Architecture (MDA). First, a technical model BPMN 2.0 is transformed into an UML class model. Then the class model is transformed into a model with Enterprise Edition (Java EE) profiles. Finally the last generated model is transformed through Meta Object Facility Script

(MOFScript) into Java EE components. The transformations are performed with Query View Transformations (QVT) Relations and MOFScript. This work contributes with transformations between profiles for generating Java EE business components related to business processes, so it helps improving the development productivity and reducing design errors.

In [BD05], UML is proposed as an effective structure for building simulation models of hybrid manufacturing systems, where machines work communicated through a network or using buffers. The model is implemented with the Matlab language [M13]. UML is used for modeling, but UML profiles are not considered.

The work in [M12] presents research results related to the main steps of the modeling and simulation process (e.g., design of simulation experiments), as well as to the application of simulation for solving different problems, inherent to operation of complex systems (e.g., analysis, optimization and management). In particular, the authors describe a tool which integrates simulation methodologies that incorporate simulation with other scientific approaches for the analysis, optimization and management of complex systems. This tool allows transforming UML models into a simulation environment; in particular, the Arena simulation environment is used.

UML has also been used in collaboration with other simulation software like DEVS [CKW10] or Petri Nets. In [KMTF08], the authors present UML diagrams that are extended with stochastic attributes. This provides possibilities for further simulation of a developed model using a simulation tool called DEVS [CKW10]. The authors generate UML models using use case and activity diagrams. On the other hand, the work in [JG07] proposes a methodology for transferring knowledge to students. The authors propose to combine the use of Petri Net and UML to model a business process as a system of discrete events. None of these papers create UML profiles.

In this paper we focus on the use of UML activity diagrams such as a pre-simulation techniques, because they allow capturing dynamic aspects of the behavior of a system. We extend the UML metamodel by defining profiles for a particular domain. Unlike previous work, like described in [TKMG08] and [BS01], our work ensures that the UML metamodel is not modified and it also meets all the semantic. Moreover, our proposed profile is specified with the formality given by the OCL [OCL12] language, allowing (1) to re-use the proposed stereotypes with any other simulation software; and (2) to re-use the simulation model with any tool that includes profiles.

3. Simulation with Arena

Modeling and simulation provide the basis for efficient solving of various problems related to the operation of complex systems like analysis, optimization and management, industry problems like mining process, etc. Simulation is considered

to be one of the most effective technologies for the analysis and planning of logistics systems.

Using simulation, you can include randomness through properly identified probability distributions taken directly from study data. For example, in this work we consider the loading and transportation of material in the field of the mining industry. In this case, there may be intersections on roads or narrow paths that only allow a truck to move at a time. Simulation also allows including the analysis and evaluation of the system when failures occurs.

Other techniques such as spreadsheet analysis or linear, goal, and dynamic programming are useful to maximize or minimize a single element (e.g., cost, utilization, or wait time). But these techniques limit the analysis to only one element, often at the expense of secondary goals. They do not allow randomness. In particular, spreadsheet analysis forces to use the average time, and will not be able to accurately capture the variability that exists in reality.

Arena simulation software [KSS08], is a general propose simulation tool enabling the construction of models over a series of modules or basic components organized hierarchically. The Arena simulation software has high level of modeling supporting graphical design. It also includes a lower level of modeling including specific details as arrival times, service times, scheduling of processes, etc.

A model is developed using modules that are part of the basic processes. In Arena, modules are the flowchart and data objects that define the process to be simulated. All information required to simulate a process is stored in modules. The dynamics associated with the processes can be viewed as nodes in a network by which entities circulate causing a change in the system state. The entities with attributes and variables compete for the services provided by the resources. Entities are items (like trucks, mineral, etc.) that are being served or produced.

Figure 1 shows a simple model build with Arena. The CREATE module is the starting point for entities in a simulation model. Entities are created using a schedule or based on a time between arrivals. Entities then leave the module to begin processing through the system. The entity type is specified in this module. The PROCESS modules are intended as the main processing method in the simulation. They include the resource by which entities compete. A resource retained by an entity must be released at some point in the model. Otherwise, a deadlock can occur. The DECIDE module allows for decision-making processes in the system. Finally, the DISPOSE modules are ending point for entities in a simulation model.

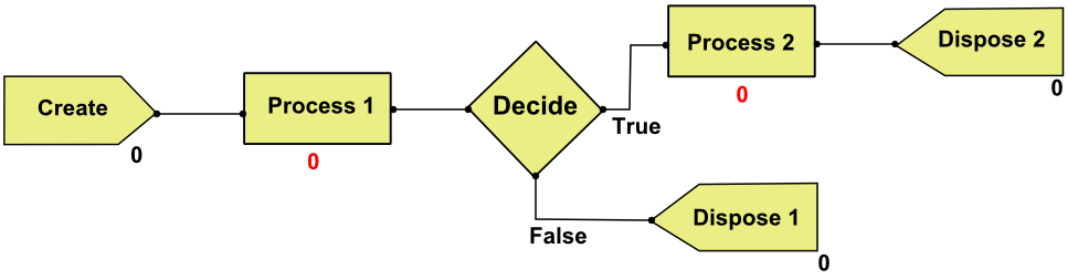

Figure 1: Basic modules used in Arena.

After the model is built, we can run simulations to obtain different metrics and statistics like resources utilization, waiting times, etc.

4. Case Study: Mining Process

A real system model is fundamental when making crucial decisions. A system model offers the possibility of planning with less error. But more importantly, it is possible to investigate new operating procedures that can be used repeatedly without interfering in the operational and daily activities of the system.

The profile proposed in this work, is applied to a system of load and transportation of minerals. During the mining process, once the rock or mineral has been removed from its extraction site, it needs to be transported immediately to its final destination. Generally, the transport system consists on a set of trucks.

Several factors are considered for the selection of trucks, being the most important factor the capacity of the vehicle in the mine. To meet daily production requirements it is necessary: a) a good performance of the equipment, i.e. to achieve the perfect match "shovel-truck" to obtain maximum extraction and transportation of minerals; and b) the road conditions should be well maintained.

The tasks performed in the daily cycle of a mining process [BL96] [BL97] start with the definition of a drilling mesh (according to height, quality and quantity of the field to be explode). Then, holes are drilled into the field. Then the explosives are introduced into the hole for blasting. The aim is to detach the rock out of the field and crush it. Then blasted material is loaded on the trucks and transported to a discharge area.

4.1 Mining Process Simulation with Arena

In this section we explain how to perform the design of load and transportation of minerals in the field of mining process using the Arena simulation tool. However, we emphasize that the focus of this paper is not the simulation design/evaluation of the mining operations but the profile design for discrete event simulations applied to the field of mining projects.

We model a transportation system for a limestone quarry. The trucks move between their loading points to their discharge points using defaults routes. The cycle begins with the release of trucks (<<Create>>) to operating fronts. Trucks are distributed among three different zones (using the module <<Decide>>) to load the mineral. The distribution percentage of the <<Decide>> module is based on the quality and quantity of material. Once in the loading zone (<<Process>>), if the blade (<<Resource>>) is free then the truck is loaded without delay. Otherwise, it should be queued (<<Queue>>) until the resource is released.

When the loading process ends, the truck is directed to the discharge zone (<<Process>>), which may be inside the plant or in the sterile area, according to the specific material been transported. This cycle is repeated according to the scheduled work.

Table 1 and 2 present time involved in a limestone quarry to move trucks from/to the treatment plant to/from the operating fronts. Trucks unload the mineral into a chute in the treatment plant. At the operating fronts we find shovels to load the trucks. Unloaded trucks move faster than loaded trucks. Then the time required to move an unload trucks from the treatment plant to the operating fronts is smaller than the time required to move the loaded trucks from the operating fronts to the treatment plant as shown in Table 1. In Table 2 we show that the time to load/unload a truck follows an exponential distribution.

Figure 2 shows the simulation design implemented with Arena. In this particular case, we evaluate the transportation system of a limestone quarry with tree operation fronts. We set the maximum number of trucks to 5. The number of trucks has been chosen to avoid long queuing times. From each operation front the trucks are loaded with minerals with different "low values", namely with different quality of the mineral. In each operation front there is a shovel in charge of putting the mineral in the truck. To decide the route for each truck we use an "N-way by chance" <<Decide>> module and we set the percentage of each route to 33%.

Table 1: Time (minutes) for mineral hauling

Route	**Shovel 1**	**Shovel 2**	**Shovel 3**
From plant to solvers	3	5	10
From solvers to plant	8	10	20

Table 2: Time (minutes) for load/unload the mineral to/from the truck

Shovel 1	**Shovel 2**	**Shovel 3**	**Treatment Plant**
Exp(6.3)	Exp(6.2)	Exp(6.4)	Exp(5.6)

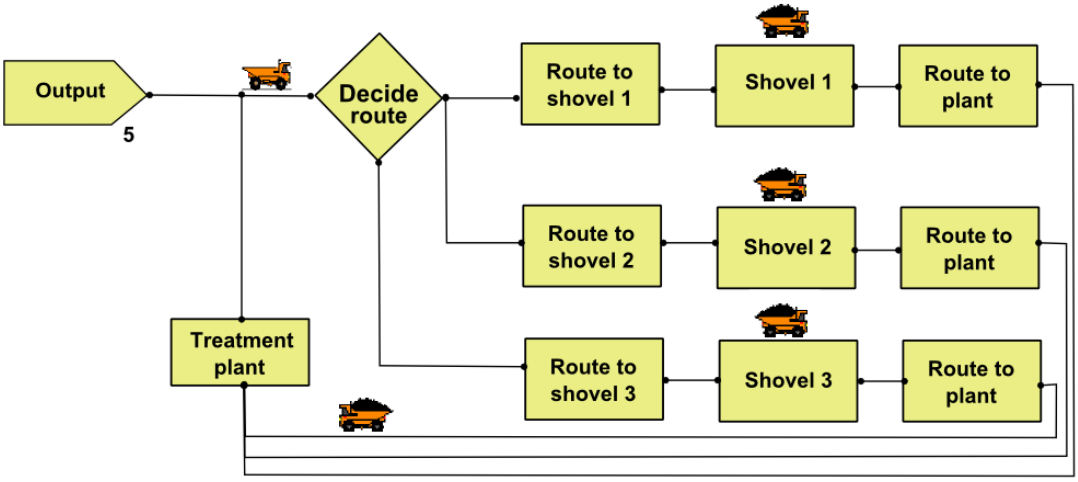

Figure 2: limestone quarry simulation.

Table 3 shows results obtained after we run the simulation for 20 days (each day has 16 worked hours). As we said before the number of trucks is set to avoid long queuing times. Therefore results show that the average number of waiting trucks in each <<Resource>> (treatment plant, shovels 1, 2 and 3) is less than 1 and the queuing time in the worst case is less than 1 minute.

Table 3: Simulation results.

Resource	**N° waiting**	**Avg. Waiting time**	**Max. Waiting Time**
Treatment plant	0.81	0.102	0.889
Shovel 1	0.06	0.024	0.729
Shovel 2	0.08	0.032	0.684
Shovel 3	0.07	0.027	0.596

5. UML Profile Definition for Mining Process Simulation

UML uses a mechanism called profile [OMG03] to adapt a model to a particular domain. A profile includes three elements: stereotypes, tagged values and constraints. Stereotypes extend the vocabulary of UML and it is possible to associate tagged values (attributes associated with extended elements) and restrictions [DGRM06], [DBRM08]. These extension mechanisms allow to adapt a particular domain to an existing metamodel.

The main builder of the profile is the stereotype, indicated as << stereotype >> [OMG03]. It has the same structure or elements (attributes, associations, operations) of the UML metamodel. So, it is not allowed to modify the semantics, structure and original concepts [APKB07] of the standard UML.

The proposed stereotypes are related to the components of the activity diagram [OMG07], because they easily adapt themselves to characterize the simulation

model [BBDL06] [F03]. This diagram already has established appropriate notations and concepts showing the steps (activities), decision points and bifurcations that occur in a transaction, so that makes it easier to display. In particular in this work we used the Rational Software Architect (RSA) tool [IBM-R] which has the advantage of supporting OCL restrictions. But any tool supporting UML profiles can be used.

The semantic analysis performed to obtain the corresponding equivalences, allows identifying the classes of the activity diagram that are necessary to extend through stereotypes. The simulation software has a greater number of modules, besides those mentioned here, some of which are identified with other classes of UML. The elements that are part of the profile are a subset of the UML metamodel applied to a simulation environment. Therefore, the profile presented in this work, extends the base classes: InitialNode, Action, DecisionNode, ActivityFinalNode, JoinNode, and ObjectNode ForkNode belonging to the activity diagram.

Although, every stereotype has its own properties, the user can edit the values of those properties. Figure 3 shows a slice of the loading-transportation cycle, which highlights the properties of the stereotype <<Create>> and its corresponding values.

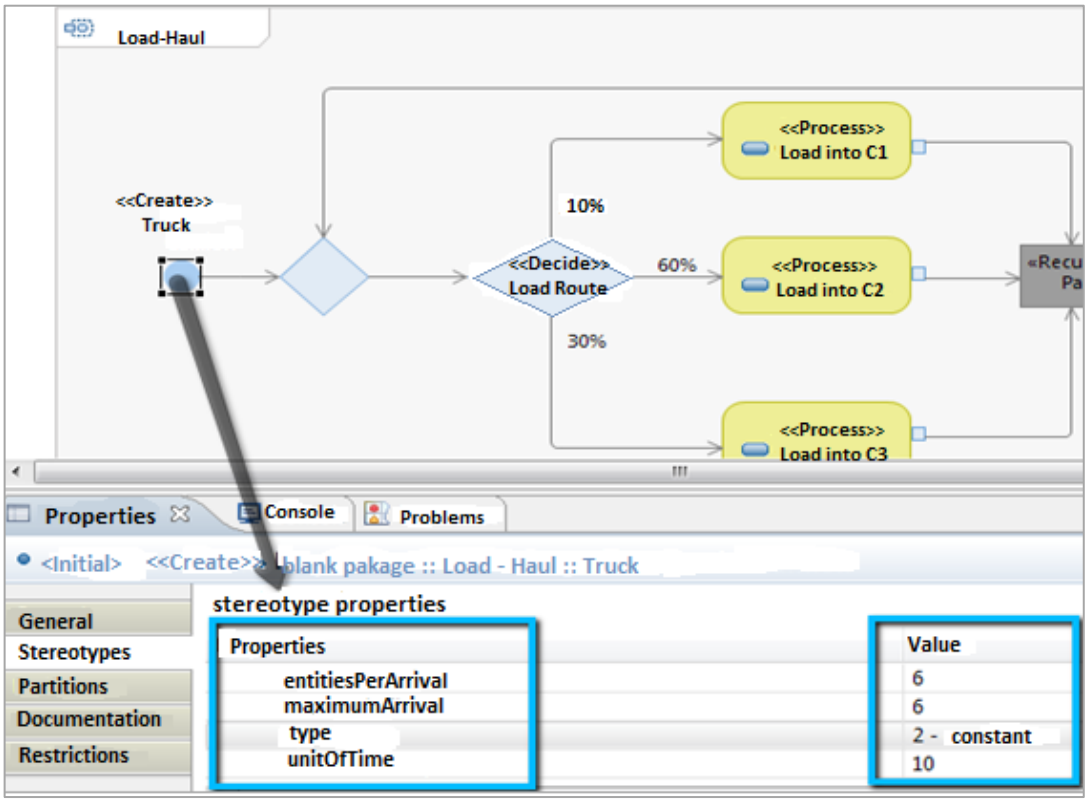

Figure 3: Properties of the stereotype <<Create>>

The stereotype <<Create>> extends the InitialNode metaclass. An InitialNode is a starting point for the implementation of an activity. In the UML specification, Superstructure v2.1.2 [OMG07], the InitialNode is defined as a generalization of a control node where an object begins to flow through the system when an activity is invoked. This package belongs to the Basic Activities defined in the abstract syntax of UML Kernel Metamodel of OMG [OMG07]. The Create module has attributes that define the input parameters, which must be defined by the user when starting the simulation. Figure 4 shows the stereotype and the extended metaclass. At the starting point of a simulation model we consider the (a) incoming flow rate generated (random or programmed constant) as well as the (b) number of entities that arrive to the system per unit time. In the mining process described in this work, every truck that enters into the system is a starting point.

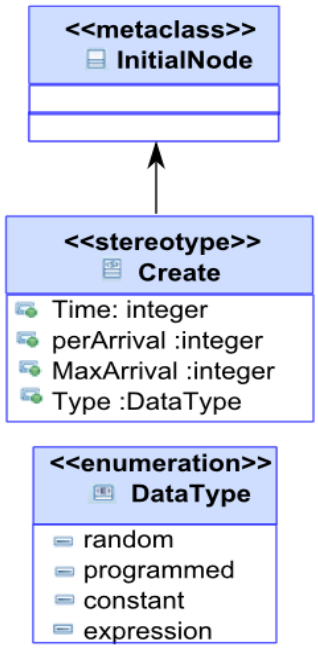

Figure 4: Relationship between metaclass and stereotype <<Create>>.

Figure 5 shows the metamodel corresponding to the OMG specification. In this figure we show the metaclass named InitialNode which is a generalization of the metaclass named ControlNode.

The stereotypes <<Process>>, <<Assign>> and <<Register>> have a base class named Action [OMG07] in the UML metamodel. For example, if we use an element of type <<Process>> in the UML model, it means that the stereotype <<Process>> behaves similarly to its base class. Additionally, it has to satisfy the semantics specified with OCL. Figure 6 shows the metaclass named <<Action>> and its corresponding stereotypes. For our particular case of study about mining processes related to load and transportation, the <<Process>> stereotype represents the load operation by means of the shovel.

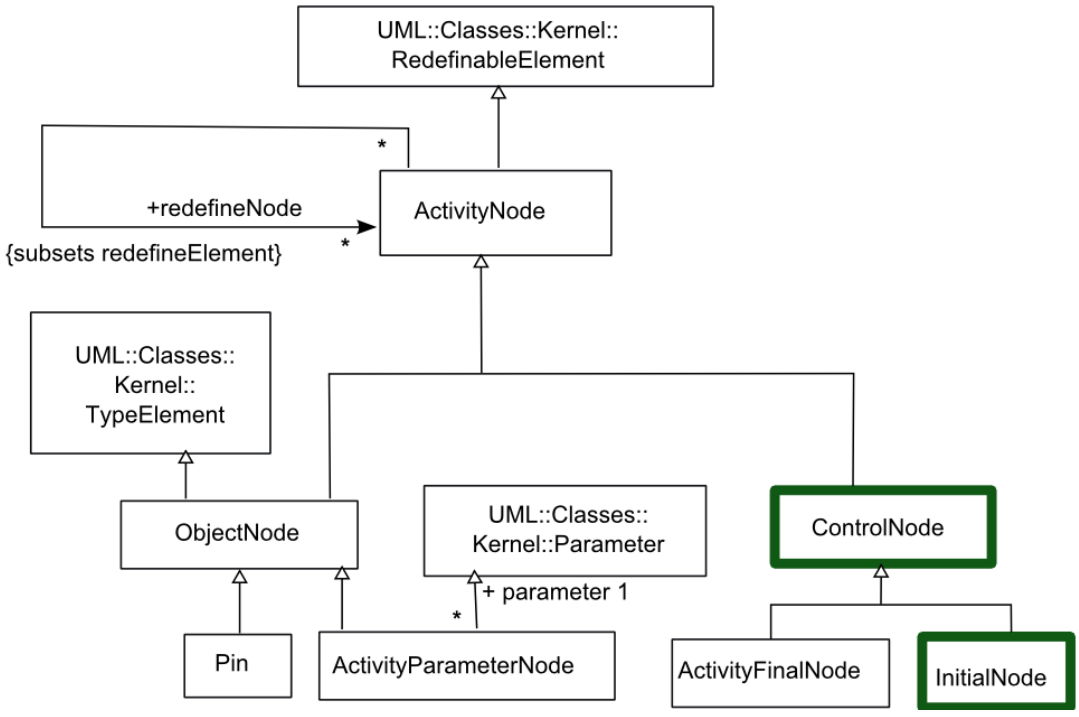

Figure 5: Metamodel of the OMG specification

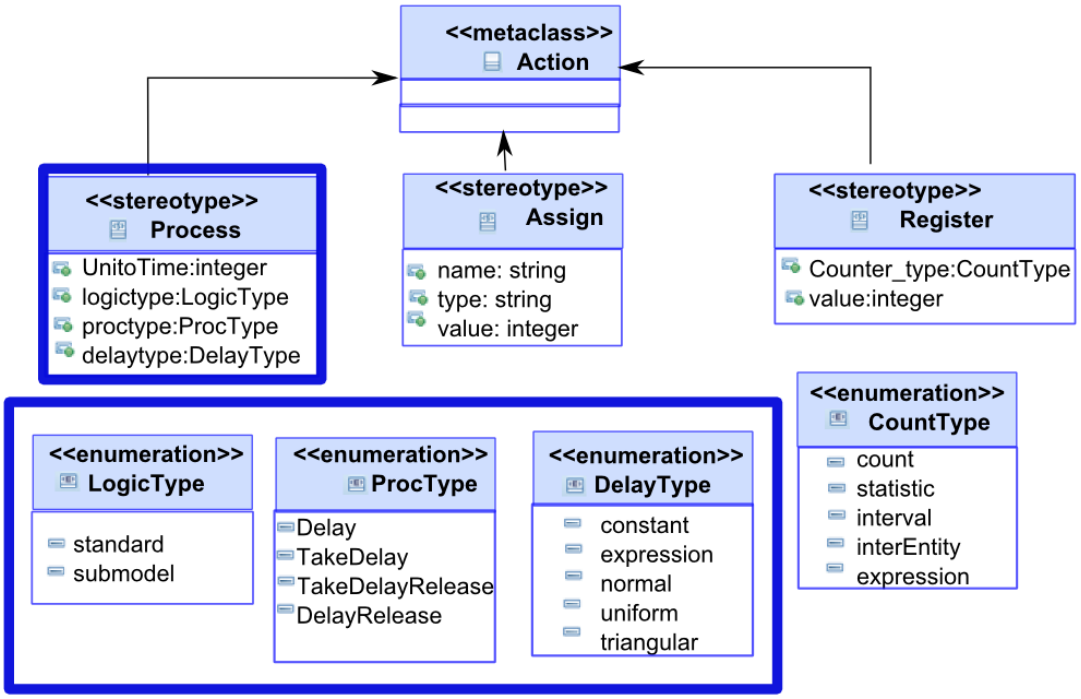

Figure 6: The metaclass Action and its stereotypes.

The UML class named JoinNode [OMG07] is extended through a so-called <<Group>> stereotype as shown in Figure 7. In a simulation model, this class should specify the number of entities to be clustered as well as how these entities will be grouped (refers to whether to use random groups or, on the contrary, the entities should be grouped according to a specific attribute).

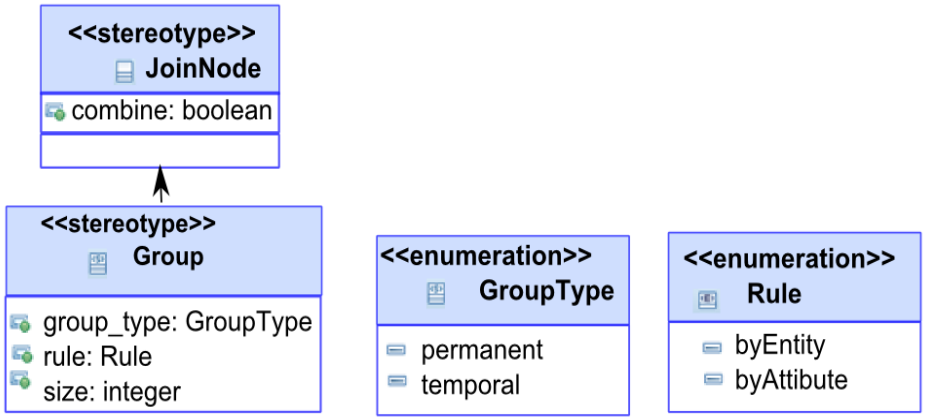

Figure 7: Metaclass JoinNode and its corresponding stereotype.

The class named ObjectNode is extended through four stereotypes <<Entity>> , <<WaitingQueue>>, <<Resource>> and <<Set>> as shown in Figure 8. The base class named ActivityFinalNode of the UML metamodel [OMG07] was extended by a stereotype called <<Dispose>>. The base class named ForkNode is extended through a stereotype called <<Divide>> [OMG07]. This is shown in Figure 9.

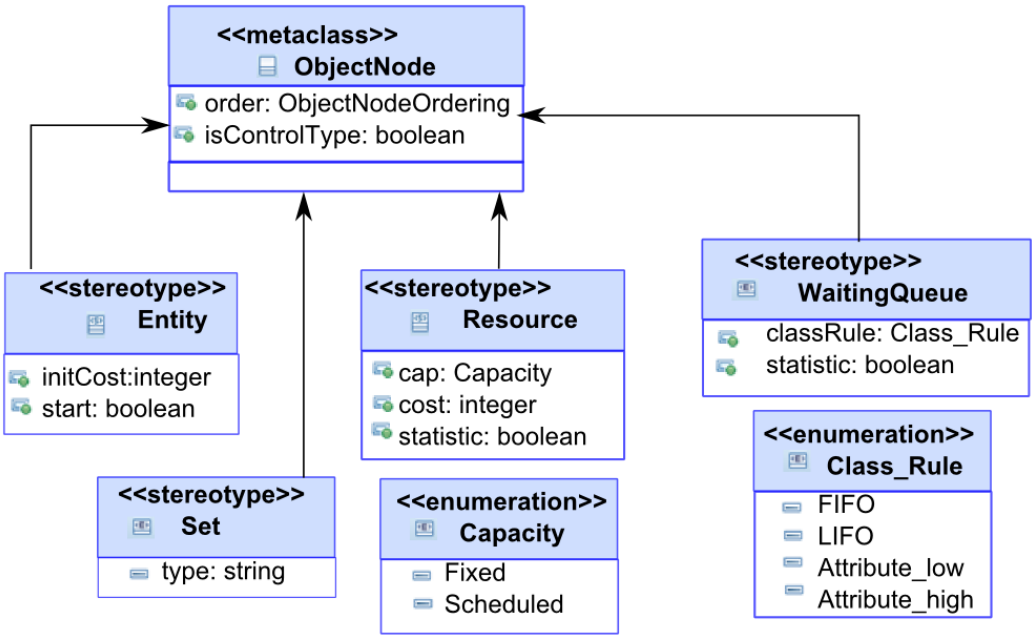

Figure 8: The ObjectNode metaclasss with its stereotypes.

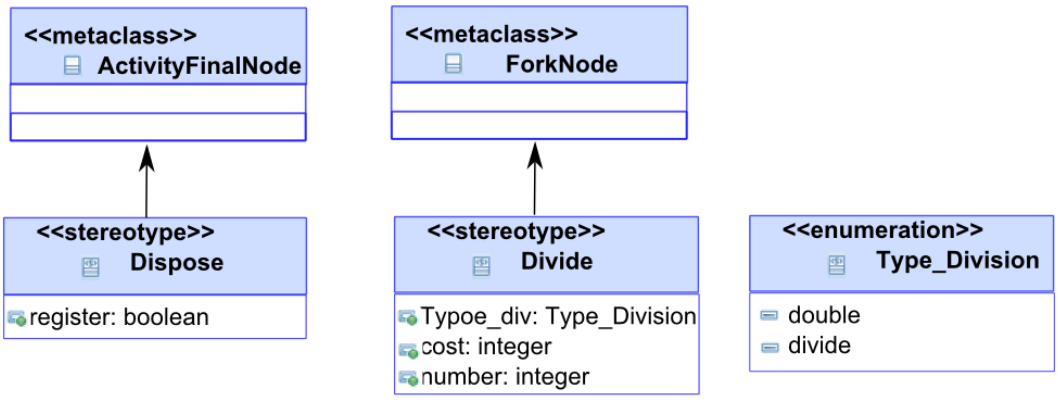

Figura 9: Metaclass ActivityFinalNode and ForkNode with its extended stereotypes.

The stereotype <<Decide>> [OMG07] extends the UML metaclass named DecisionNode. Figure 10 shows this relationship.

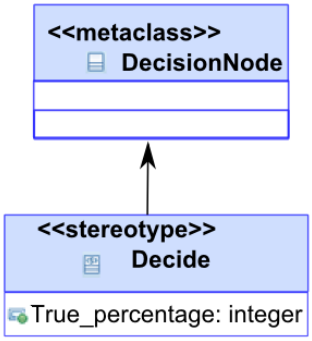

Figure 10: The metaclasss DecisionNode and its stereotype.

The basic modules that integrate the simulation model of Arena are related to each other through connectors [KSS08]. These connectors indicate how entities flow within the model. These connectors are represented with the metaclass of the activity diagram named ControlFlow. It is not necessary to extend this base class because there is a bilateral matching of the semantics.

Stereotypes defined above, integrate the simulation profile that is applied in a mining process. The extended metaclasses was obtained as a result of observing the behavior of the basic modules of the simulation process. This analysis showed that while they share similarities in semantics, these modules have specific attributes that are not specified in the UML base classes, which is why we extended those classes, as described in the previous paragraphs.

5.1 OCL Restrictions

A model is an abstraction of a system. In particular in this work we model a mining process. It should be as accurate as possible. In order to find an unambiguous and consistent model, it is necessary to reformulate the restrictions to the developed profile.

The language proposed by UML to specify the restrictions of the diagrams, is the so-called OCL (Object Constraint Language) [OCL12]. OCL offers a convenient way of getting models with a high degree of consistency, because it is a formal language used to describe keywords in the model [OCL06], making them more accurate and less ambiguous.

OCL is fully integrated into UML. We can check the values of the model elements and set restrictions on them. An example of a restriction on the profile element, in this case the stereotype <<Decide>>, is as follows:

```
Context Decide inv:
((self.incoming.source->forAll(a | a.oclIsTypeOf(Decide)) or
(a.oclIsTypeOf(Process))or (a.oclIsTypeOf(Divide)) or
(a.oclIsTypeOf(Group)))and
(self.outgoing.target->forAll(a | a.oclIsTypeOf(Decide)) or
(a.oclIsTypeOf(Dispose))or (a.oclIsTypeOf(Process)) or
(a.oclIsTypeOf(Divide))or (a.oclIsTypeOf(Group))))
```

The restrictions fulfill the purpose of validating the model where the proposed stereotypes apply. If an error occurs, it is immediately observed. In this case, we can introduce corrections so that the model complies with the pre-established conditions and reconciling with the expected results.

6 Conclusions

UML allows expressing specific concepts of discrete event simulation in the field of mining, through the extension of its basic classes. In particular, simulation software modules like CREATE and PROCESS are easily identified with the elements of the activity diagram. This feature provides high functionality to the proposed extended model, because its behavior is similar to the simulated system. System validation which was obtained by applying the profile to the mining process, where the load-transport cycle is a key part of a reservoir, satisfied the requirements imposed by the daily production.

Having a standard language like UML implies that it is possible to obtain a model with stereotypes with well-formed rules without ambiguous concepts. In this paper we presented an extension of UML for the basics elements of the Arena simulation

software, which provides a higher level of abstraction. The integrated vision of these two languages (UML and simulation) helps to build a consistent model for the process been simulated. Therefore, combining both tools facilitates the development of a system.

The proposed profile presents advantages at two levels:

Metamodel level: At this level, any UML tool supporting profiles can be used to import the mining profile defined in this work. In this way, any engineer can use any UML tool for build their mining models using the same modeling language (UML profile). The second advantage is that if using a simulation language, other than Arena, or if the same Arena is extended with new characteristics, those characteristics can be easily included into the profile. The engineers use a modeling language (UML profile) independent of the simulation language (Arena, Simula, GPSS, etc.). The mining UML profile can be included with the specific semantic of any simulation language. Finally, using formal languages supporting mathematic specifications such as OCL (first order logic, theory of set and theory of bag) ensures much more accurate models than the ones used by other traditional modeling languages such as ERD (Entity Relationship Diagram), DFD (Data Flow Diagram) as general modeling language or ExtendSim Suite, Arena Blocks as specific modeling languages. The mathematics specifications are incorporated as OCL specifications in the meta-modeling level of the profile. Therefore, the proposed UML profile can be re-used with any UML tool that includes profile definitions with OCL.

Model level: At this level, the engineer builds its standard models using the proposed profile, and these models can be used in any mining environment. The stereotypes can be applied with any mining process simulation software (not only Arena). Moreover, any UML tool that allows incorporating the standard profiles according to the OMG can be used to modify these models. Finally, we obtained models verified mathematically with OCL. Those are more accurate models according to the restrictions that are specified in the profile level.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this article.

References

[APKB07] Abdullah M.S., Paige R., Kimble C., Benest I.: A UML Profile for Knowledge-Based Systems Modelling, Fifth International Conference on Software Engineering Research, Management and Applications. IEEE, 2007.

[BBDL06] Bartolini C., Bertolino A., De Angelis G., Lipari G.: A UML Profile and a Methodology for Real-Time Systems Design, Proceedings of the 32nd EUHOMICHO Conference on Software Engineering and Advanced Applications. IEEE, 2006.

[BCN96] Banks J., Carson J. S., Nelson B. L., Discrete-event system simulation. Edición 2nd ed. 1996

Upper Saddle River, New Jersey Prentice Hall, 1996.

[BD05] Barbarisi O., Del Vecchio C.: UML Simulation Model for Hybrid Manufacturing Systems, Proceedings of the13thMediterranean Conference on Control and Automation Limassol, Cyprus, June 27-29, 2005.

[BL96] Bustillo Revuelta M.; López Jimeno C., Recursos Minerales. Tipología, Prospección, Evaluación, Explotación, Mineralurgia, Impacto Ambiental. Madrid, 1996.

[BL97] Bustillo Revuelta M.; López Jimeno C., Manual de Evaluación y Diseño de Explotaciones Mineras. Madrid, 1997.

[BS01] Barjis J., Shishkov B.: UML based Business Systems Modeling and Simulation, the 4th International EUROSIM Congress, June 26-29, 2001.

[CKW10] Castro R., Kofman E., Wainer G., A Formal Framework for Stochastic DEVS Modeling and Simulation, In Simulation: Transactions of the Society for Modeling and Simulation International, volume 86, 2010.

[CWM] Object Management Group. Common Warehouse Metamodel (CWM) Specification. OMG document, ad/2001-02-01. 2001

[DBRM08] Debnath N.C., Baigorria L., Riesco D., Montejano G.: Metrics applied to Aspect Oriented Design using UML profiles, Computers and Communications 2008, Symposium on Digital Object Identifier, pp. 654-657, IEEE, 2008.

[DGRM06] Debnath N.C., Garis A., Riesco D., Montejano G.: Defining Patterns Using UML Profiles, IEEE Conference Proceeding, 2006.

[DMZRM12] Debnath, N.; Martinez, C.A.; Zorzan, F.; Riesco, D.; Montejano, G., "Transformation of business process models BPMN 2.0 into components of the Java business platform," Industrial Informatics (INDIN), 2012 10th IEEE International Conference on , vol., no., pp.1035,1040, 25-27 July 2012

[DBRM08] Debnath, N.; Baigorria, L.; Riesco, D.; Montejano, G., "Metrics applied to Aspect Oriented Design using UML profiles," Computers and Communications, 2008. ISCC 2008. IEEE Symposium on , vol., no., pp.654,657, 6-9 July 2008

[EHHS00] Engels G., Hausmann J.H., Heckel R. and Sauer S., Dynamic meta modeling: A graphical approach to the operational semantics of behavioural diagrams in UML. In Proceedings of UML, volume 1939. LNCS, 2000

[E10] Elmer - CSC". CSC — IT Center for Science Ltd. Retrieved 2010-06-24.

[F03] Frankel D.S.: Model Driven Arhcitecture. Applying MDA to Enterprise Computing, 2003.

[FSL13] The Facsimile Simulation Library. http://facsim.org/. Facsimile copyright © 2004-2013 Michael J Allen

[IBM-R] IBM Rational Software Architect version 8.5. Copyright IBM Corporation 1987, 2012. http://www-01.ibm.com/software/rational/.

[JG07] Jan Pels H., Goossenaerts J.: Modelling Technique for Discrete Event Simulation of Operational Processes, IFIP International Federation for Information Processing, Volume 246 pp. 305-312, 2007.

[KMTF08] Kleins A., Merkuryev Y., Teilans A., Filonix M., A meta-model based approach to UML modelling and simulation. Proceedings of the 7th WSEAS International Conference on System Science and Simulation in Engineering (ICOSSSE '08), 2008.

[KSS08] Kelton W.D., Sadoswski R.P., Sturrock D.T.: Simulación con Software Arena, Cuarta Edición, McGraw-Hill, 2008.

[FV04] Lidia Fuentes-Fernández and Antonio Vallecillo-Moreno. An Introduction to UML Profiles. The European Journal for the Informatics Professional. Vol 2, Nro. 2. Pp:6-13- 2004.

[M12] Merkuryev Y., Riga Technical University. The Modelling and Simulation of Complex Systems: Methodology and Practice. An Overview. Information Technology and Management Science. 2012.

[M13] Matlab® http://www.mathworks.com/products/matlab/. © 1994-2013 The MathWorks, Inc.

[MM03] Miller J. and Mukerji J., MDA guide. Technical report, Object Management Group, 2003. http://www.omg.org/mda.

[NST09] M. N. Nashalji, M. A.i Shoorehdeli and M. Teshnehlab. Fault Detection of The Tennessee Eastman Process Using Improved PCA and Neural Classifier. International Journal of Electrical & Computer Sciences IJECS-IJENS Vol.09 No:09 pp:54-59. 2009

[OCL06] UML 2.0 OCL Specification, http://www.omg.org, August 2006.

[OCL12] OMG Object Constraint Languaje (OCL), OMG Document Number: formal/2012-01-01.

Standard document URL: http://www.omg.org/spec/OCL/2.3.1. January 2012.

[OMG03] OMG Unified Modeling Languaje (OMG UML), Infrastructure, Version2.3, OMG Document Number: formal/2010-05-03, www.omg.org/spec/UML/2.3.

[OMG07] OMG Unified Modeling Languaje (OMG UML), Superestructure, Version2.1.2, OMG Document Number: formal/2007-11-02, Standard Document, http://www.omg.org/spec/UML/2.1.2/Superstructure/PDF, 2007.

[S12] SimPy http://simpy.readthedocs.org/en/latest/. © Copyright 2002 -2012 Team SimPy.

[TKMG08] Teilans A., Kleins A., Merkuryev Y., Grinbergs A.: Design of UML models and their simulation using ARENA, WSEAS Transactions on Computer Research, Issue 1, Volume 3, January, 2008.

[W08] Watson A.: Visual Modeling: past, present and future, White paper UML Resource Page, 2008.

[YDHZ12] S. Yin, S. X. Ding, A. Haghani, H. Hao, P. Zhang. A comparison study of basic data-driven fault diagnosis and process monitoring methods on the benchmark Tennessee Eastman process. Journal of Process Control. Volume 22, Issue 9, October 2012, Pages 1567‑1581.

[YLD13] S. Yin, H. Luo and S.X. Ding. Real-Time Implementation of Fault-Tolerant Control Systems With Performance Optimization. IEEE Transactions on Industrial Electronics (Volume:61, Issue: 5) pages: 2402 – 2411. 2013