COMPARISON OF SIMIO AND ARENA SIMULATION TOOLS

António Vieira[(a)], Luís Dias[(b)] , Guilherme Pereira[(b)], José Oliveira[(b)]

[(a) (b)] University of Minho, Campus Gualtar, 4710-057, Braga, Portugal.

[(a)] antonio6vieira@gmail.com, [(b)] luis.dias / guilherme.pereira / jose.oliveira @algoritmi.uminho.pt

KEYWORDS: Simulation, Arena, Simio

ABSTRACT

The purpose of this paper is to establish a comparison between Simio and Arena, helping a vast community of simulation practitioners to gain access to advanced modelling capabilities to address complex problems. Several aspects were compared, such as: concept of simulation models, animation development, modelling philosophies, Simio libraries and Arena templates, concept of entities, interface of the tools and Simio objects versus Arena blocks. The comparison was consolidated through the analysis of two case studies where the authors aimed to emphasize the way each simulation tool addresses some important issues related to model construction. The several compared aspects indicate the many advantages of using the more recent tool. Thus, this object-oriented tool appears to have all the conditions to trigger a widespread paradigm shift in the way practitioners build models.

1. INTRODUCTION

Simulation modelling is being widely used for performance improvement of many systems (Hlupic, 2000, Hlupic and Paul, 1999). Consequently, the number of simulation tools available is also increasing and tools comparison becomes a required task. Yet, "most of scientific works related to tools comparison/reviews analyse only a small set of tools and usually evaluating several parameters separately avoiding to make a final judgement due to the subjective nature of such task" (Dias et al., 2007). One of such works was developed by Hlupic and Paul (1999). The authors compared a set of simulation tools, distinguishing between users of software for educational purpose and users in industry. In his turn, Hlupic (2000) developed "a survey of academic and industrial users on the use of simulation software, which was carried out in order to discover how the users are satisfied with the simulation software they use and how this software could be further improved". Dias and Pereira et al. (2007, 2011) compared a set of tools based on popularity on the internet, scientific publications, WSC (Winter Simulation Conference), social networks and other sources. "Popularity should never be used alone otherwise new tools, better than existing ones would never get market place" (Dias et al., 2007). However, a positive correlation may exist between popularity and quality, since the best tools have a greater chance of being more popular. The author's final classification indicated that the most popular tool at the date was Arena. The only new tool on the ranking was Simio which obtained a good classification, meaning that this tool has good odds of becoming more popular and widely used in the future.

The development of Simio and Arena simulation tools was leaded by the same author: Dennis Pegden. Thus, it is normal that there are some resemblances between them. "However care is required to ensure that your knowledge of Arena does not become a handicap that prevents exploiting the full power of Simio" (Pegden, 2013a). Dennis Pegden (2007) exposed a comparison between the two tools with the purpose to "help experienced Arena users transition from Arena to the new Simio" (Pegden, 2013a). His white paper focused mainly on exposing some differences in the concepts of the two tools regarding: models, entities and resources concepts, animation development and modelling philosophies. Yet, some practical differences from the two tools were not addressed. To that end, this paper intends to compare both tools, taking into account several relevant factors but also enlightening the learning of Simio.

Chapter 2 presents a review over the analysed literature. In chapter 3, a comparison over the two simulation tools is made, regarding: general concepts, interface of the tools and Simio objects and Arena blocks. Lastly, two case studies are presented and analysed in chapter 4. In the final chapter, the main conclusions of the conducted work are drawn.

2. LITERATURE REVIEW

2.1. Arena

In 1993 SIMAN and CINEMA (simulation languages) were combined into a single tool: Arena (http://www.erlang.com.br/arena.asp). This tool is a simulation environment consisting of module templates, built around SIMAN language constructs, as well as other facilities and the CINEMA animation package (Altiok and Melamed, 2010). Thus, when an Arena model is created it is implemented in SIMAN code which is then compiled and run without any need to write programming code. SIMAN consists of blocks and elements. Blocks are basic logic constructs that represent operations (e.g. seize block). Elements are objects that represent facilities such as resources, queues and tallies (Altiok and Melamed, 2010). In 1995 the first version of Arena for Windows 95 was released. It was the first to run in 32-bit systems. From 2000 on, after being acquired by Rockwell, the software received a huge development boost and new versions, in

increasingly shorter time periods, were launched. Nowadays, the software is considered the most popular simulation tool in the world (Dias et al., 2007, Pereira et al., 2011). Since Arena already has many years of existence, a lot of published documents is available (Altiok and Melamed, 2010, Kelton et al., 2002). However, the same does not apply to Simio.

2.2. Simio

Simio, developed in 2007 (Vik et al., 2010), is based on intelligent objects (Sturrock and Pegden, 2010, Pegden, 2007, Pegden and Sturrock, 2011). These "are built by modellers and then may be used in multiple modelling projects. Objects can be stored in libraries and easily shared" (Pegden, 2013b). Unlike other object-oriented systems, in Simio there is no need to write any programming code, since the process of creating a new object is completely graphic (Pegden and Sturrock, 2011, Pegden, 2007, Sturrock and Pegden, 2010). The activity of building an object in Simio is identical to the activity of building a model. In fact there is no difference between an object and a model (Pegden, 2007, Pegden and Sturrock, 2011). A vehicle, a customer or any other agent of a system are examples of possible objects and, combining several of these, one can represent the components of the system in analysis. Thus, a Simio model "looks" like the real system (Pegden and Sturrock, 2011, Pegden, 2007). This fact can be very useful, particularly while presenting the results to someone non-familiar to the concepts of simulation.

In Simio, the model logic and animation are built in a single step (Pegden and Sturrock, 2011, Pegden, 2007). This feature is very important, because it makes the modelling process very intuitive (Pegden and Sturrock, 2011). Moreover, the animation can also be useful to reflect the changing state of the object (Pegden, 2007). In addition to the usual 2D animation, Simio also supports 3D animation as a natural part of the modelling process (Sturrock and Pegden, 2010). To switch between 2D and 3D views the user only needs to press the 2 and 3 keys of the keyboard (Sturrock and Pegden, 2010). Moreover, Simio provides a direct link to Google Warehouse, a library of graphic symbols for animating 3D objects (Sturrock and Pegden, 2010, Pegden and Sturrock, 2011).

Simio offers two basic modes for executing models: the interactive and the experimental modes. In the first it is possible to watch the animated model execute, which is useful for building and validating the model. In the second, it is possible to define one or more properties of the model that can be changed, in order to see the impact on the system performance (Sturrock and Pegden, 2010).

According to Pegden (2007): although Simio incorporates a number of innovative features in pursuit of this goal, "only time will tell if this tool has bridged the many practical issues that must be addressed to trigger a widespread paradigm shift in the way practitioners build models" (Pegden, 2007).

Currently there are not many studies that use Simio for modelling systems. Even so, it is possible to find some studies that used this tool for other types of problems. Akhtar et al. (2011) studied the role of consanguineous marriages in causing congenital defects. Li and Wang (2011) developed a micro simulation model to evaluate the performance and service level of a ticket office. Vik et al. (2010) used Simio to model a logistic system design of a cement plant. Brown and Sturrock (2009) also used this tool to improve a set of production processes. Lastly, Kai et al. (2011) used Simio to explore simulation of casualty treatment in wartime.

3. COMPARISON OF THE TOOLS

This chapter first concentrates on different views of the conceptual philosophy of both tools when developing a simulation model. Thereafter, it explains different approaches of both tools as far as the interface with the user is concerned. Lastly, this chapter shows how the behaviour of Simio objects could be addressed with Arena blocks - and this corresponds to the implementation issue, where the authors aim to analyse the practical aspects of building simulation models.

3.1. General Concepts Comparison

- **Simulation model concept:** In Arena when a user refers to his "model" he is referring to the Arena simulation model. Yet, in Simio, a model is simply an object that can be instantiated in other models.
- **Animation development:** In Arena a user animates his model as a 2-step process: first he draws process flows for the model, and then, in a separate area of the same drawing space, he adds levels, animated routes and others, that are linked back to the process flow (Pegden, 2013a). On the contrary, in Simio, the user drags objects to a drawing space. Since they represent the physical components of the system (Pegden and Sturrock, 2011, Pegden, 2007) the objects define both the logic and the animation of the model. Thus, modelling and animation are done as a single step (Pegden, 2013a). Moreover, Simio provides a direct link to Google Warehouse, a library of graphic symbols for animating 3D objects (Sturrock and Pegden, 2010, Pegden and Sturrock, 2011).
- **Modelling philosophies:** In Arena, a model is built by using the process orientation paradigm. In this philosophy, the user defines elements that hold the state of the system, and build process flows using blocks that perform actions on the elements. These blocks are passive and are only activated by the arrival of an entity (Pegden, 2013a). On the other hand, Simio is a multi-paradigm modelling tool, in the sense that it supports both object orientation and process orientation. In fact, the ability to mix object-based and process modelling within the same model is one of the unique and very powerful features of Simio (Pegden, 2013a). This way, users can use the faster capacities of the object paradigm and the more flexible capacities of the process orientation (Pegden, 2013a). Thus, a user builds object-

based models by thinking in terms of the physical objects in the system (machines, conveyors, etc.). These objects are placed in the Facility and interact with each other based on their internal logic. The process orientation is used in the Process panel in which the user constructs Arena-like process flows (Pegden, 2013a). This processes are used to either customize the behaviour of an existent object, or to create new object definitions (Pegden, 2013a). However, there are some differences between the terminology of the processes in Simio and in Arena, in the sense that in the first a process is comprised of steps, elements, and tokens that flow through a process executing steps that alter the state of one or more elements. Hence, steps are like Arena blocks (Pegden, 2013a).

- **Libraries versus templates:** An Arena template is a set of hierarchical blocks (modules) that can be placed in process logic. In contrast, a Simio library is a collection of object definitions for placing objects in the facility. This defines a new library that can be used by other models, and so on. Thus, Simio libraries and Arena template panels share the basic notion of user-defined library, but they differ in the modelling orientation that they are designed to support (Pegden, 2013a).
- **Entities concept:** Entities in Arena are part of a model of the system and their only purpose is to carry information (attributes) and to execute a process. In fact every entity in an Arena model must have exactly the same attributes (Pegden, 2013a). In Simio, entities have object definitions, thus have their own intelligent behaviour and can make decisions, such as reject requests, decide to take a rest, etc. Moreover, each entity has a token that corresponds to it and executes a process. Thus, an entity in Arena corresponds to a token of Simio (Pegden, 2013a).

3.2. Interface

3.2.1. Arena

Arena possesses 2 other tools incorporated: Input analyser and Output analyser. While the first fits a distribution to a sample data, the latter is a tool for analysing the data resulted from a simulation process.

There are 3 main regions that can be identified in the main Arena window:

1) Project bar

Located on the left side, it contains several templates that can be attached or detached to the Project bar. Templates are a set of blocks pre-defined or user-defined, i.e., a collection of modelling tools. From those, basic process, advanced transfer and advanced process stand out. More information on Arena templates can be found at (Vieira, 2013).

To build a simulation model with Arena a user needs to use modules from the above mentioned templates. There are 2 types of modules: Flowchart blocks and data modules (Garrido, 2009). The user can places blocks on the model window and connect them to form a flowchart that describes the system he is modelling. Data modules are data in spreadsheet-like format that enables the user to edit some information.

2) Model window flowchart view

This region is located on the right side of Arena. It is the workspace for the simulation model and will contain all the model graphics, flowcharts, animation, and other drawings.

3) Model window spread sheet view

Located on the right-hand side and below the flowchart view, it shows the model data and some details of the blocks being used/selected.

3.2.2. Simio

In Simio there are 3 areas that are always visible and can be seen in Figure 1: the ribbons, the browse panel and the tabbed panel views. These areas are described in more detail in (Vieira, 2013).

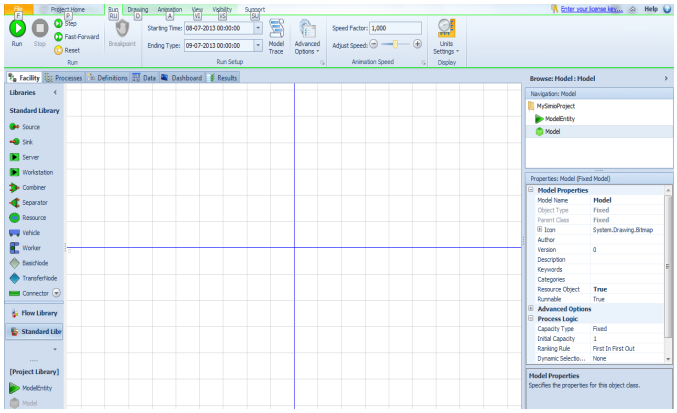

Figure 1: General view of Facility

The tabbed areas are divided in: Facility, Process, Definitions, Data and Results. More information on the panels can be found at (Vieira, 2013).

3.3. Objects versus Blocks

This comparison will be made by trying to model the same behaviour of Simio objects of the Standard library, resorting to some Arena blocks. The objects belonging to the Standard library are:

Source: This object is responsible for creating entities. Figure 2 illustrates a Source object.

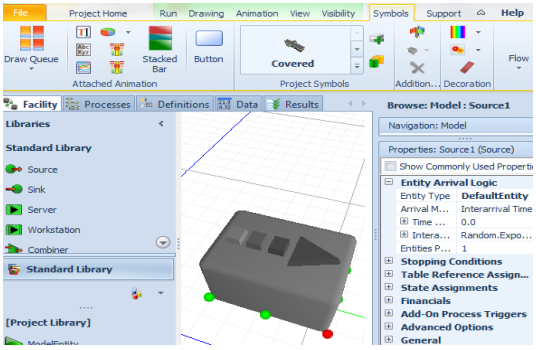

Figure 2: Source object

This object is comprised by an output node, an output buffer queue and the object itself. It is possible to establish a comparison between this object and the Create block of Arena, since both can define the entity type, Interarrival times, number of entities per arrival,

maximum number of arrivals and the time offset until the first arrival. However, in Simio it is possible to edit many more properties like assign values to states, assign add-on processes, make table references and state assignments, change animation of created entities and others. Figure 3 represents a Create block of Arena.

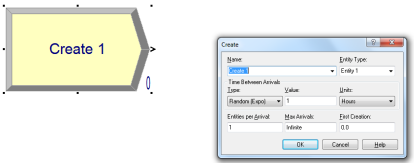

Figure 3: Create block

Sink: Figure 4 illustrates a Sink object. This object is responsible for eliminating entities from the system.

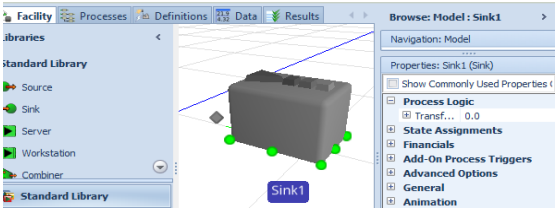

Figure 4: Sink object

This object is comprised by an input node, an input buffer queue and the object itself. The Dispose block of Arena has the same goal of this object. The Simio object also allows the definition of add-on processes, state assignments and others. Figure 5 displays a Dispose block.

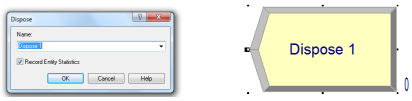

Figure 5: Dispose block

Server: "The Server object can be used to model a single server or a single processing center with multiple identical servers, depending upon the capacity specified for the processing station" (www.simio.com). Figure 6 shows a Server object which is comprised by an input node, an input buffer queue, a processing station, an output buffer queue, an output node and the object itself.

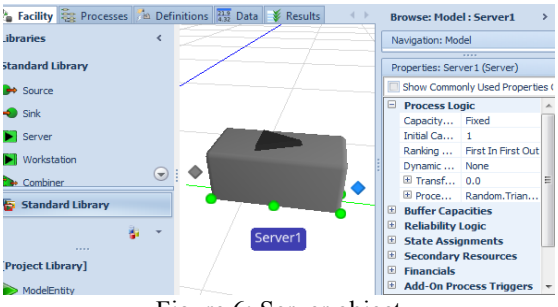

Figure 6: Server object

The Process block of Arena has the same objective of this object, since both model processes with a determined processing time. In a Process block, the user has to specify the type of process (e.g. seize, seize delay release), the allocated resource, the processing time and needs to assign the resource type. In a Server, this is done in a more natural way, since the user does not need to assign a specific type of resource to be seized. Despite having the same goal, the Simio object allows a user to edit a superior set of properties like secondary resources, failures, state assignments, add-on processes, and others. Figure 7 illustrates a Process block.

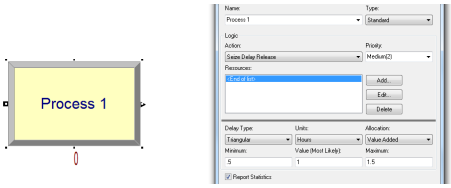

Figure 7: Process block

Workstation: The most complex object of the Standard Library. It is similar to a Server except that it models the processing station in far more detail, since the latter is represented by an operation divided into 3 activities: setup, processing, and teardown. All the entities moving through the Workstation will perform each of these activities. Figure 8 illustrates a Workstation object. It is not possible to establish a comparison between this object and a single block of Arena. In order to model this object, a great number of blocks would have to be used.

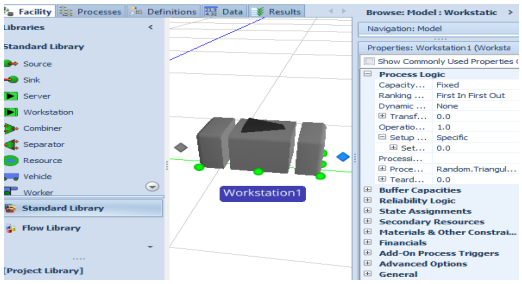

Figure 8: Workstation object

Combiner: Matches multiple entities, groups them into a batch, and then attaches the batched members to a parent entity. Figure 9 illustrates a Combiner.

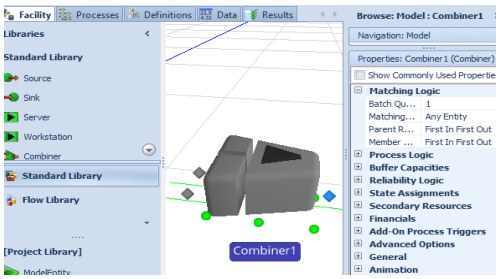

Figure 9: Combiner object

This object is comprised by a parent input node, a member input node, a parent input buffer queue, a member input buffer queue, a processing queue, an output buffer queue, an output node and the object itself. In Arena it is possible to model this behaviour by using the blocks Match to synchronize the entities and then, the Batch block to attach those entities together. Nevertheless, in Simio there is the possibility to visualize the member entities of a parent entity as they travel through the model. Figure 10 illustrates the

addressed situation. In this image a tray (parent entity) waits for 2 cakes (member entities) to be combined. The cakes can be visualized through the creation of a batch member queue for the tray entity.

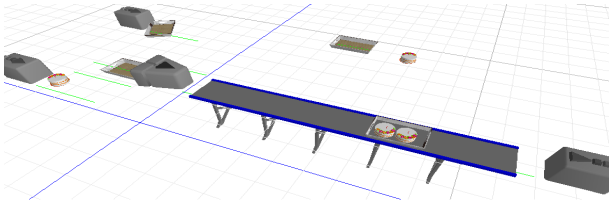

Figure 10: Combiner example in Simio

The Combiner also offers the possibility to edit properties like failures definition, add-on processes, state assignments, secondary resources, capacity types and others. Figure 11 displays the two blocks: Batch and Match of Arena.

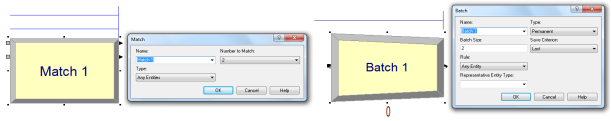

Figure 11: Match and batch blocks

Separator: Either separates batched members from a parent entity, or makes copies of an entity. Figure 12 illustrates a Separator object, which is comprised by a main object, an input node, an input buffer queue, a Processing queue, a parent output buffer queue and a member output buffer queue.

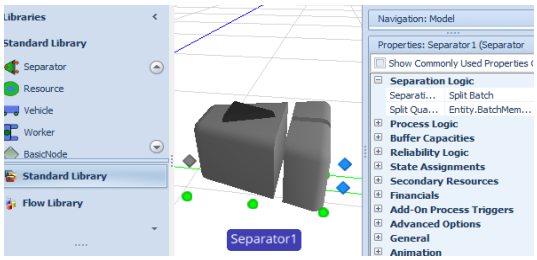

Figure 12: Separator object

A comparison between this object and the Separate block of Arena can be established. Apart from the common goal of the object and the block, the first also offers the possibility of editing further properties like define failures, add-on processes, state assignments, secondary resources, capacity types and others. Figure 13 illustrates a Separate Arena block.

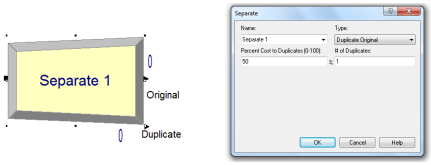

Figure 13: Separate block

Resource: Generic object with capacity that can be seized and released by other objects. Entities do not pass through this object, unlike the previous ones. In fact, the placement of this object on the Facility, only intends to declare the existence of a resource type that can be seized and released. In Arena, a user defines resource types through the Resource data module, in order to achieve the same behaviour. Despite this, in Simio the concept of resources is quite different and much more robust than in Arena, due to the fact that any object can seize and release any other object. In this object, a user can also define failures, add-on processes among others. Figure 14 shows a Resource object.

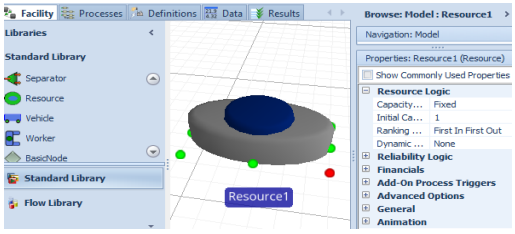

Figure 14: Resource object

Vehicle: Used to model devices that follow a fixed route (e.g. bus, train, etc.), or respond to dynamic requests for pickups (e.g. taxi, AGV, etc.). Similarly to the Resource object, the placement of this object on the Facility only intends to declare the existence of a vehicle type. Therefore this object is not connected to any object and thus entities do not pass through it. Figure 15 illustrates a Vehicle object.

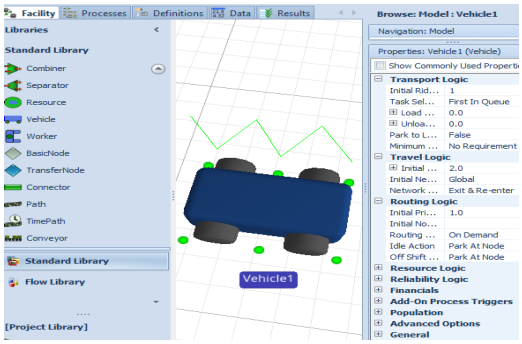

Figure 15: Vehicle object

This object is comprised by a ride station queue and the object itself. Arena also possesses the concept of vehicles to transport entities though these cannot follow fixed routes. Additionally, in Arena a transporter can only transport an entity at a time, whilst in Simio this can be done several entities at a time. To model transports in Arena it is necessary to use the template AdvancedTransfer, more specifically the blocks Request, Transport and Free. This object also allows a user to define failures, population number and others. Figure 16 illustrates these blocks combined, in order to model a transport.

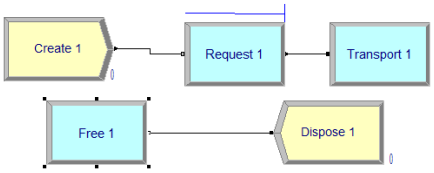

Figure 16: Using Arena blocks to model a transport

Worker: Defines a moveable resource that may be seized, released or used to transport entities between nodes. In contrast to the Vehicle that supports a Routing

Type which can be on demand or fixed, the Worker always operates on demand, i.e., the Worker always waits for either a visit request or a transport request. Additionally, unlike the Vehicle, the Worker has the ability to follow a work schedule and the Worker always assigns priority to seize visit requests over transport requests. This object is comprised by a ride station queue and the object itself. In Arena, the notion of moveable resources does not exist. Figure 17 shows a Worker object.

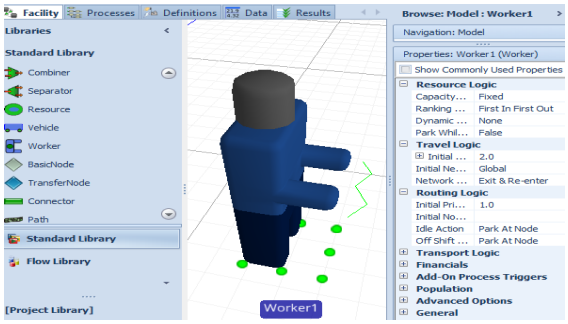

Figure 17: Worker object

BasicNode: Models a simple intersection between multiple links and can also be used as input nodes of objects. This object cannot model changes of destination. In Arena, a block with a similar concept to the nodes of Simio is the Station block. Figure 18 shows a BasicNode object and Figure 19 illustrates a Station block.

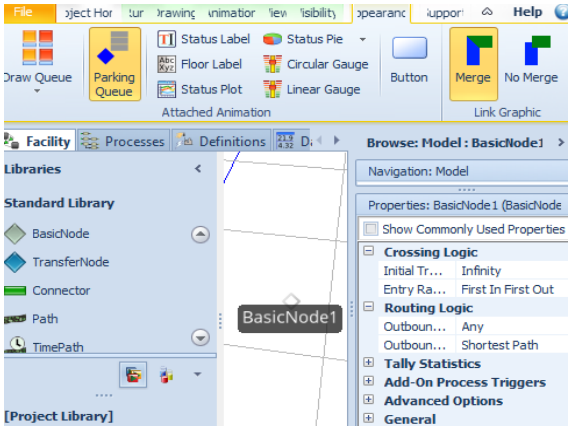

Figure 18: BasicNode object

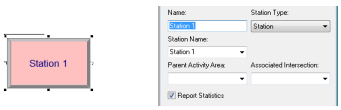

Figure 19: Station block

TransferNode: Models a complex intersection for travel mode. Unlike the previous object, this can model changes of destination. Additionally, the TransferNode can be used as output nodes of objects. Figure 20 illustrates a TransferNode object.

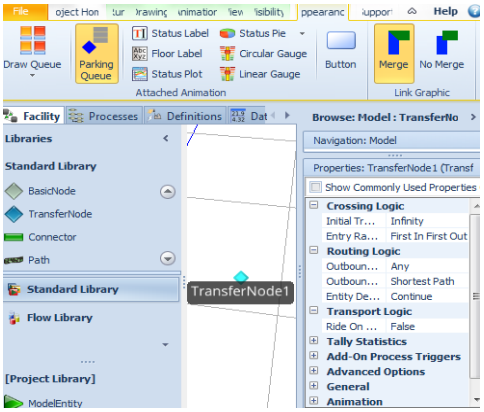

Figure 20: TransferNode object

Connector: Represents a simple zero-time travel link between 2 nodes. In Arena, the same goal is achieved by connecting 2 blocks, using the connect option. Figure 21 displays this object connecting a Source to a Sink.

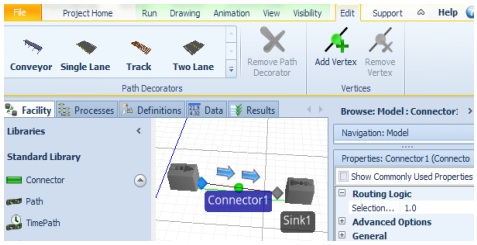

Figure 21: Connector object

TimePath: Used to transfer entities between 2 nodes with a specified travel time. Figure 22 shows a TimePath connecting a Source to a Sink.

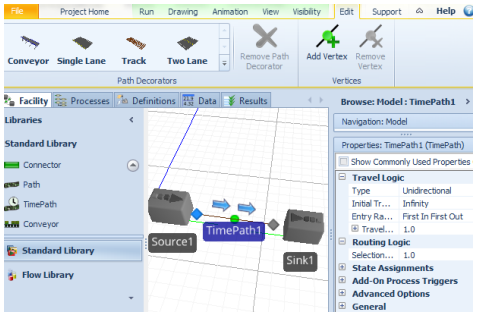

Figure 22: TimePath object

In Arena, to model the same behaviour, it is necessary to use the template AdvancedTransfer, more specifically, the blocks Station and Route. Figure 23 illustrates an example of these blocks being used.

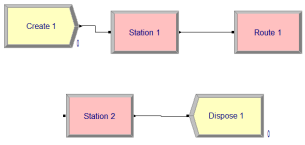

Figure 23: Using Arena blocks to model routes

Path: Represents links over which entities may move independently, at their own speed rates. Figure 24 shows a Path object, connecting a Source to a Sink. In Arena it is not possible to achieve the same behaviour of this object.

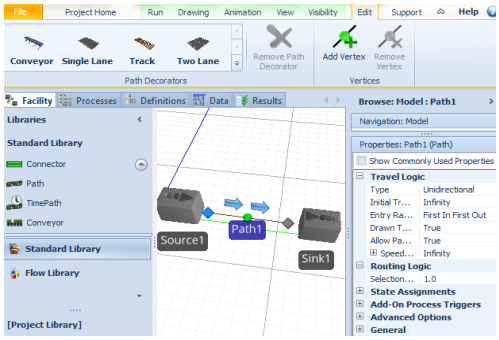

Figure 24: Path object

Conveyor: Entities traveling on this kind of connection do not "move". Their movement is done by a conveyor that can be accumulating or non-accumulating. Figure 25 illustrates a Conveyor between a Source and a Sink.

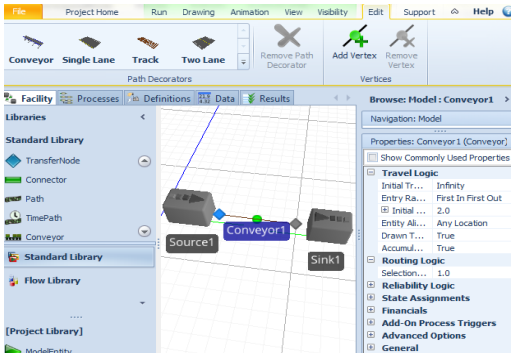

Figure 25: Conveyor object

In Arena, to model the same behaviour it is necessary to use the AdvancedTransfer template, more specifically, the blocks Station, Access and Convey. Figure 26 illustrates a conveyor being modelled in Arena.

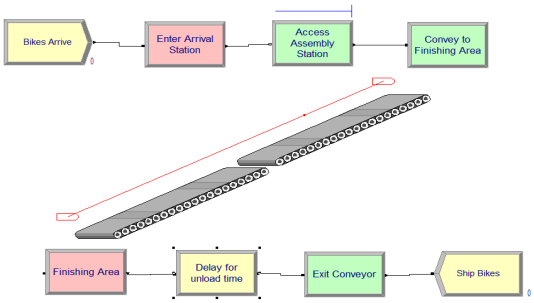

Figure 26: Using Arena blocks to model conveyors

4. CASE STUDIES

This chapter intends to introduce two case studies and analyse the way both simulation tools address the same problem. First, using a basic example - then, adapting it to the use of transports, which is a very important aspect of the development of any simulation model regarding the representation of a real operating system.

4.1.1. Basic Problem

- **Problem description:** This problem consists on a situation where trucks arrive at a factory and need to unload its merchandise. Each truck is loaded with malfunctioning TVs that have to be repaired by repairmans. After a TV is repaired, it has to go to the inspection, where the inspectors will evaluate its condition. If the TV has no more defects, it goes to a truck parked outside. The trucks wait for repaired TVs and then leave the system.
- **Basic problem in Arena:** This system was modelled using Arena. Figure 27 shows the developed simulation model.

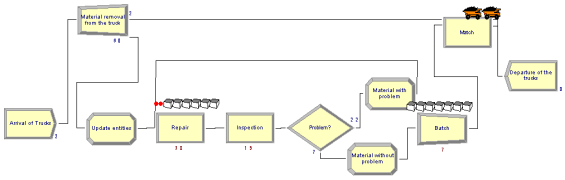

Figure 27: Basic problem in Arena

For modelling this system, two types of entities were used: one to represent the trucks and another to represent its merchandise. To create entity types, a user needs to use the Data module Entity. The Create block "Arrival of trucks" only creates entities of the type truck. After the creation of a truck, the Separate block "Material removal from the truck" is responsible for separating the truck and its merchandise, though, the merchandise will be animated with the same image as the truck. Since this does not correspond to what is intended, the Assign block "Update entities" updates the image of the entity of the merchandise. After being separated from its merchandise, the truck waits for the repaired TVs on the Match block. Thus, the TVs need to be repaired by the Process "Repair". In this process, an entity seizes a repairman, delays the allocation for a specified process time correspondent to the time needed to repair a TV and then releases the repairman, allowing his allocation to another entity. When the reparation is concluded, the TV follows to the "Inspection" process. In it, an inspector is seized by an entity for a specified process time and then is released, in order for the inspector to be allocated to another entity. This process evaluates whether or not the TV still has any defects. Nearly 25% of the entities that pass through this inspection fail on the test and thus, need to be repaired again. This situation is modelled by the block Decide "Problem?". Before repeating the "Repair" process, the image of a red ball is assigned to these entities. In the Batch block, the fixed TVs that will go to a truck wait until a specified value of TVs is reached. After the number is reached, the truck leaves the system with the merchandise, through the Dispose block.

- **Basic problem in Simio:** Figure 28 shows the developed model in Simio.

By looking at the model developed in Simio, the most notable difference to the same model developed in Arena is the animation. Simio's model is much more realistic, in fact, it "looks" like the real system. However, there are several more differences that will now be addressed. Firstly, in Simio a user may associate an array of symbols to an object. Thus, in this model

two symbols were associated to the TV: a regular TV for TVs that haven't been repaired or TVs with no problems and a red TV for TVs with defects. Secondly, the creation of an entity type is done by simply dragging a ModelEntity object to the Facility. In this case, two of these objects were dragged: the truck and the TV. Another difference is the fact that in Arena, in order to model the change of destination of TVs that need to be repaired anew, the block Decide needs to be used. Yet, in Simio, the same goal can be achieved by adding a Path between the output node of the "Inspection" and the input node of the "Repair" and editing the respective probabilities of each destination. Additionally, any object can perform state assignments when entities enter or before leaving the object. In this sense, when entities fail the inspection and need to be repaired again, the Path connecting the two objects assigns the symbol red TV to the image of the entity. Lastly, in Simio only the Combiner object is necessary to model the Match and Batch blocks. In fact, this makes much sense, since the two blocks are used together almost every time in Arena. The parent input buffer queue displays the trucks waiting for the merchandise and the member input buffer queue displays the TVs waiting to be combined. When the later reaches a specified number, the first truck of the parent input buffer queue leaves the system through the Sink.

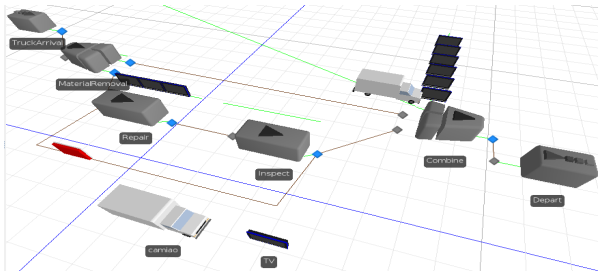

Figure 28: Basic problem in Simio

4.1.2. Problem with Transports

- **Problem description:** This problem corresponds to the previous one with the introduction of transports. More specifically, in this case study, the merchandise removal from the trucks that arrive at the system and the loading of the trucks with repaired TVs is done with the help of forklifts. Also, when a TV needs to be inspected or to repeat the reparation process, it is placed on a conveyor.
- **Problem with transports in Arena:** Figure 29 illustrates the developed model in Arena and Figure 30 shows the correspondent model animation.

As can be seen, the physical model is created separately from the logical model.

- **Problem with transports in Simio:** Figure 31 shows the developed model in Simio.

As can be seen, the same model can be built using a lower number of components. This is due to the fact that, for instance, to model the Vehicles (forklifts) and the Conveyors it is only necessary to drag the correspondent objects to the Facility and edit their properties. On the other hand, in Arena a great number of blocks need to be used to achieve the same goal.

To develop the considered case studies in both tools, the authors concluded the modelling in Simio was done in a simpler, faster and more intuitive way. Regarding the animation of the models, by examining the figures related to the case studies, it is clear that Simio models "look" like the real system. Lastly, there are some systems that can be modelled in Simio and are impracticable to achieve in Arena. Such a case can be consulted in (Vieira, 2013).

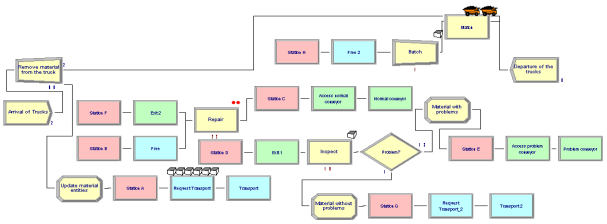

Figure 29: Problem with transports in Arena

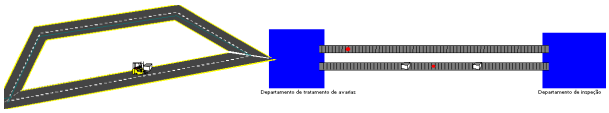

Figure 30: Animation of the problem with transports

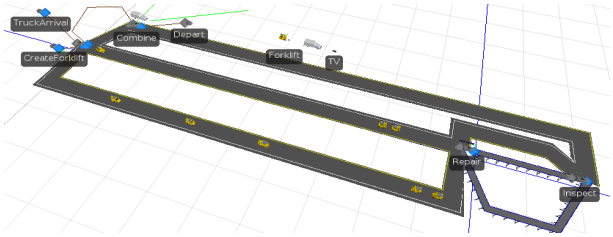

Figure 31: Problem with transports in Simio

5. CONCLUSIONS

Simulation modelling is being widely used for performance improvement of many systems (Hlupic, 2000, Hlupic and Paul, 1999). Hence, the comparison of simulation tools becomes a required task. In this context, Dias et al. (2007, 2011) compared a set of tools based on the popularity of most commercial tools. The author's final classification indicated that the most popular tool at the date was Arena and the only new tool on the "top 20" was Simio, both developed by the same authors: Dennis Pegden and David Sturrock. Thus, it should be normal that there are some resemblances between them. In this context, this paper intended to compare both tools taking into account several factors, such as: concepts of simulation model, animation development, modelling philosophies, Simio libraries versus Arena templates, entities concept, interface of the tools and Simio objects versus Arena blocks. Lastly, two case studies were addressed, in order to analyse the way both problems should be modelled on each tool.

Once the comparison of the tools was concluded, several aspects can be highlighted. Firstly, the Arena interface is simpler than Simio's. Regarding published documentation, Arena is highly more referenced than Simio. However, the latter is a much more recent tool.

The most visible difference between the models of Arena and Simio is the animation. Whilst in Arena the animation is developed in a separated step of the modelling, in Simio the modelling and the animation are done as a single step and the direct link with Google Warehouse makes the models very similar to the real systems. Even so, there are more differences in the systems modelling approach of each tool. Namely, Arena uses the process orientation while Simio is a multi-paradigm tool and its main feature is the ability to model intelligent objects and everything in Simio is an object. Consequently, Simio's entities are objects with their own intelligent behaviour and can make decisions, reject requests, etc. Moreover, entities have tokens that correspond to them and execute processes, while in Arena the only purposes of entities is to carry information (in their attributes) and to execute processes (Pegden, 2013a). Thus, an entity in Arena is similar to a token in Simio (Pegden, 2013a).

The comparison of objects in the Standard library of Simio with the blocks in the Basic Process template of Arena showed some resemblances between them, in the sense that most Arena blocks can be modelled by a correspondent Simio object, making it a very intuitive tool to use for Arena users. However, there are cases, in which the user needs to use a great number of blocks to achieve the same goal of a single Simio object. However, a great number of advantages of the Simio objects are noted, like the possibility of assigning values to states, add-on processes, arrays of animation, failures and schedules without the need to create new objects. Furthermore, some cases can be easily and quickly modelled in Simio than in Arena, for instance the utilization of conveyors or transports in Simio is achieved by simply dragging the correspondent objects to the Facility while in Arena a greater number of blocks need to be used.

The development of the chosen case studies showed that it is possible to model the same model of Arena, in Simio. Nevertheless, it required less effort and time to model the considered examples, in Simio. Moreover, Simio model's shape mimics the real system's layout, having required a lower abstraction level to develop.

An example of a case study developed in Simio, where it would be impracticable to model the same system in Arena was also given.

The compared aspects indicate the many advantages of using Simio. However, there are some down-sides typical of a recent tool. For instance, in Simio a user cannot create a clock to have a better perspective of the simulation time passing. In Arena this is very simple to add to a model. Another feature that Arena possesses and Simio does not is the incorporated tools: Input and Output Analyser. Also, at the date this paper was prepared there were some features that were not completely implemented (e.g. acceleration of entities). Nevertheless, this object oriented tool appears to have all the conditions to "trigger a widespread paradigm shift in the way practitioners build models" (Pegden, 2007).

ACKNOWLEDGEMENTS

This work has been supported by FCT – *Fundação para a Ciência e Tecnologia* in the scope of the project: PEst-OE/EEI/UI0319/2014.

REFERENCES

AKHTAR, N., NIAZI, M., MUSTAFA, F. & HUSSAIN, A. 2011. A discrete event system specification (DEVS)-based model of consanguinity. *Journal of Theoretical Biology,* 285**,** 103-112.

ALTIOK, T. & MELAMED, B. 2010. *Simulation Modeling and Analysis with ARENA*, Elsevier Science.

BROWN, J. E. & STURROCK, D. 2009. Identifying Cost Reduction and Performance Improvement Opportunities Through Simulation. Proceedings of the 2009 Winter Simulation Conference: M. D. Rossetti, R. R. Hill, B. Johansson, A. Dunkin and R. G. Ingalls, eds.

DIAS, L., PEREIRA, G. & RODRIGUES, G. 2007. A Shortlist of the Most Popular Discrete Simulation Tools. *Simulation News Europe,* 17**,** 33-36.

GARRIDO, J. M. 2009. *Object Oriented Simulation: A Modeling and Programming Perspective*, Springer-Verlag.

HLUPIC, V. Simulation software: an Operational Research Society survey of academic and industrial users. Simulation Conference, 2000. Proceedings. Winter, 2000 2000. 1676-1683 vol.2.

HLUPIC, V. & PAUL, R. 1999. Guidelines for selection of manufacturing simulation software. *IIE Transactions,* 31**,** 21-29.

KAI, Z., RUICHANG, W., JIE, N., XIAOFENG, Z. & HAIJIAN, D. Using Simio for wartime casualty treatment simulation. IT in Medicine and Education (ITME), 2011 International Symposium on, 9-11 Dec. 2011 2011. 322-325.

KELTON, W. D., SADOWSKI, R. P. & SADOWSKI, D. A. 2002. *Simulation with Arena*, McGraw-Hill School Education Group.

LI, J. & WANG, L. Microscopic simulation on ticket office of large scale railway passenger station. Advanced Forum on Transportation of China (AFTC 2011), 7th, 22-22 Oct. 2011 2011. 41-47.

PEGDEN, C. D. Simio: A new simulation system based on intelligent objects. Simulation Conference, 2007 Winter, 9-12 Dec. 2007 2007. 2293-2300.

PEGDEN, C. D. 2013a. An Introduction to Simio for Arena Users. Simio. White paper. Available online at: http://www.simio.com/resources/white-papers/For-Arena-Users/An-Introduction-to-Simio-For-Arena-Users.htm.

PEGDEN, C. D. 2013b. Intelligent objects: the future of simulation. Simio. White paper. Available online at: http://www.simio.com/resources/white-papers/Intelligen-objects/Intelligent-Objects-The-Future-of-Simulation-Page-1.htm.

PEGDEN, C. D. & STURROCK, D. T. Introduction to Simio. Proceedings - Winter Simulation Conference, 2011 Phoenix, AZ. 29-38.

PEREIRA, G., DIAS, L., VIK, P. & OLIVEIRA, J. A. 2011. Discrete simulation tools ranking: a commercial software packages comparison based on popularity.

STURROCK, D. T. & PEGDEN, C. D. Recent innovations in Simio. Proceedings - Winter Simulation Conference, 2010 Baltimore, MD. 21-31.

VIEIRA, A. 2013. *Master Thesis. Micro simulation to evaluate the impact of introducing pre-signals in traffic intersections.* Department of Production and Systems - University of Minho.

VIK, P., DIAS, L., PEREIRA, G., JOS, #233, OLIVEIRA & ABREU, R. 2010. Using simio for the specification of an integrated automated weighing solution in a cement plant. *Proceedings of the Winter Simulation Conference.* Baltimore, Maryland: Winter Simulation Conference.