Early Validation Framework for Critical and Complex Process-Centric Systems

Fahad Rafique Golra, Joël Champeau, Ciprian Teodorov

Early Validation Framework for Critical and Complex Process-Centric Systems

Fahad Rafique Golra(✉), Joël Champeau, and Ciprian Teodorov

Lab STICC UMR6285, ENSTA Bretagne, Brest, France
{fahad.golra,joel.champeau,ciprian.teodorov}@ensta-bretagne.fr

Abstract. Guaranteeing the correctness of the future system is of vital importance for the development of critical and complex systems. Rigorous software development methodologies are used for such systems, where formal methods for the verification of properties guarantee the required level of correctness. For process-centric, critical and complex systems, one needs continuous observation of the process (through simulation and visualization) both during the development for correctness and afterwards for process improvement. We present a framework with associated methodology and tools, for the development of process-centric critical and complex systems. This early validation methodology promotes formal verification of the process model alongside agent-oriented simulation and visualization of the process models in a distributed context. Moreover, the process simulation technique proposed in the methodology allows step-wise replacement of the simulated components with the actual system services. We explain the proposed methodology using an adaptation of a real-life case-study from the military sector.

Keywords: Business process · Modeling · Simulation · Visualization

1 Introduction

As the demand for web based solutions and services is rising, we witness an increased focus on business services and componentization of business functionalities. Many organizations have shifted their focus from a data-centric approach to a process-centric one for information system technology and solutions [15]. Initially, the focus remained on workflow technology and was limited to the automation of routine processes and the execution of relatively simple activities. Gradually process-centric systems have moved towards coordination and collaboration management and information based decision making activities that can manage complex, dynamic and higher value mission critical processes [15]. When process definitions serve as the main controller of the enterprise systems, we need to use a development methodology that ensures that the defined processes exhibit the necessary properties for running the system. This promotes early validation of the processes used in such systems. In case of critical and complex systems, one needs to guarantee that the process, around which the complete system is

developed, is rigorously tested for correctness using formal methods. Apart from model checking approaches, one also needs to simulate and visualize the process, when the system under development is safety/mission-critical. In such systems it is important to simulate the impact of individual components on the collective behavior of the system. Simulation of scenarios generates accurate information about the utilization, performance and overall effect. Visualizations can then be used for analysis, discovering sensitivities, optimization and monitoring [12].

In the context of process-centric critical and complex systems, we suggest using iterative development and incorporating model checking techniques alongside process simulation and visualization approaches. However, such a methodology will have to face the challenges of seamless integration of these varying approaches that have different point of views over the same components of the system. The novelty of our approach lies in the integration of these approaches under a holistic methodology. The goal is to tackle the challenges that we face in the integration of an evolutionary development process using a model checking technique and agent-based simulation. Our research objectives are:

- *Objective 1: Simulating and visualizing the higher abstraction level business processes.* Process simulations for software systems require low level implementation details which can be considered as noise by a business user. We focus on a methodology to hide this complexity under multiple abstraction levels that are linked together through mappings.
- *Objective 2: Formal verification of a process model according to the operational semantics defined in the process interpreter.* Traditionally translation of a model into a specific formal language for verification guarantees the correctness of the model in that particular language. It raises the problem of ensuring the equivalence between the semantics of the (generated) implementation from the model and the semantics defined in the formal language *e.g.* Promela [11]. The goal is to verify the process with the semantics defined in the interpreter that would later become the process engine of a deployed process-centric system.
- *Objective 3: Guaranteeing the correctness for both the individual activities and the complete process model containing them.* To ensure the correctness of a process model, we argue that one needs to decompose the verification problem into: the verification of the actions described within an individual activity and the verification of the control flow specified in the process model containing these individual activities. Simulation and visualization of the process activities helps in the verification of actions defined in an individual activity. Model checking techniques focus on the verification of certain properties in a control-intensive process model.
- *Objective 4: Allowing seamless transfer from simulation to the actual system services.* The transition of a system from a simulation to an actual system should be seamless. We focus on a framework (methodology, architecture and associated tools) that allows the transition from simulated components to actual components for the development of a critical and complex system.

With these objectives in mind, we propose a framework for process development and early validation that offers an architecture, a methodology and associated tools developed for putting this methodology into practice. The rest of the paper is organized as follows: In Sect. 2, we explain the existing technologies used in the framework and present the proposed framework. In Sect. 3, we present an adaptation of a real-life case study to explain this framework. We discuss the related works in Sect. 4 and finally conclude the paper in Sect. 5.

2 Process-Centric Critical and Complex Systems

We propose a framework that relies on the use and integration of some existing technologies, coupled with dedicated components and tools. The most notable of these technologies are the NATO Architecture Framework (NAFv4) for enterprise modeling and the DirectSim framework for simulations.

NATO Architecture Framework (NAFv4): It provides a standardized way to develop and describe architectures for both military and business use [3]. The framework is defined through multiple viewpoints distributed across five layers: Concepts, Service, Logical, Physical Resource and Architecture Meta-Data. NAFv4 suggests using ArchiMate specification [2] as the standard metamodel. The behavior in NAFv4 is captured by the following viewpoints around processes, states and sequences.

- *C4: Standard Processes* viewpoint identifies the business activities and links them to corresponding capabilities.
- *S4: Service Functions* viewpoint identifies the functions performed by each service.
- *L4: Logical Activities* viewpoint identifies the logical activities, their grouping and composition and the logical flow between them. It forms the business level process model.
- *P4: Resource Functions* viewpoint specifies the functionality of resources in the architecture. These functions and the flow (of data and control) between them form the application process model.
- *L4-P4: Activity to Function Mapping* viewpoint specifies the mapping between the activities of the business process and the functions of the application process. It also addresses the relation between the functions of the application process and that of business services.

DirectSim: DirectSim[1] is an open source framework, developed by the French ministry of armed forces, that allows the development of simulation applications. It has mostly been used for military simulations [4], but being open source, it is increasingly being used for other civilian projects as well. The simulation engine at the core of the framework is based on a timed agent-based event simulation.

[1] https://www.directsim.fr.

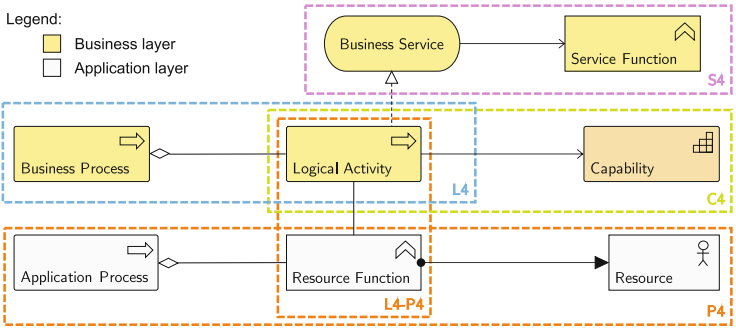

Fig. 1. Archimate diagram of the process related viewpoints of NAFv4

A monitor and a set of plug-ins allow the edition and visualization of scenarios in 2D and 3D spaces. Multiple views of the tool allow the interaction with the simulation *e.g. scenario editor view* to edit the scenarios, *default view* for the visualization, *etc.* It also offers a test toolkit for testing the scenarios. A compilation of the simulation may be necessary, in case one wants to modify the behavior of the simulation. The core simulation framework is based on .Net framework and the development is done in C#.

2.1 Proposed Framework

We propose a framework for the development and early validation of processes in critical and complex systems through formal verification, simulation and visualization of the process models. Apart from the existing technologies that we used, all other tools that we developed for this framework are accessible online[2].

Process Modeling: We follow NATO architecture framework (NAFv4) for enterprise modeling, where processes are described at different levels of abstraction. In an ArchiMate model presented in Fig. 1, we extract the process relevant viewpoints and show their interrelation using the concerned NAFv4 concepts. The *C4: Standard Processes* viewpoint of NAFv4 identifies the list of business activities that are eventually modeled in the Business layer. The functions perform by each business service are also defined in the same Business layer through the *S4: Service Functions* viewpoint. Hence, taking *L4: Logical Activities* viewpoint into account gives us access to the complete business process. The *P4: Resource Functions* viewpoint is used to capture the application processes. We use the *L4-P4: Activity to Function Mapping* viewpoint of NAFv4 to map the application process activities to the corresponding business process activities. This mapping between the two levels of abstraction can be used either to create a holistic process view that contains the activities from both the abstraction

[2] https://github.com/plug-obp.

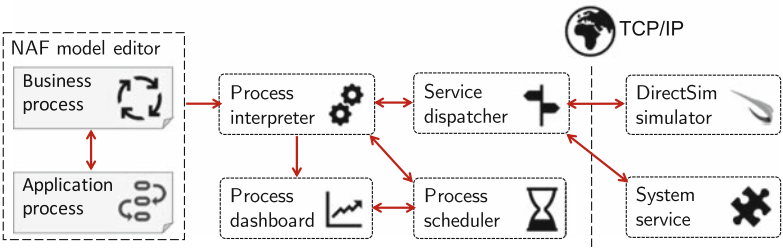

Fig. 2. Architecture of the proposed framework

levels, or to trace back to the business level activities from the application level activities. In accordance with the first objective outlined in Sect. 1, we use the latter approach, where we interpret the application process in way that individual activities can be traced back to the business process. This allows us to simulate business processes by passing through the application processes that carry the details needed for the simulation.

Process Interpretation: Our approach is to interpret the process model before model checking and simulation activities, as shown in Fig. 2. We have developed a process interpreter than takes the serialized version of the process model (*.bpmn) as input. Similar to other proposals in the literature (*e.g.* [5]), our process interpreter uses direct semantics, given in terms of features and constructs of the process model, rather than in terms of their low-level encoding into another formalism. It takes into account the core elements like activity nodes and other control nodes *i.e.* initial, final, decision, merge, fork and join nodes. This process interpreter also takes into account multiple pools with participants, each having a separate process. The collaboration between multiple pools is handled by interpreting the message flows between them. The interpreter develops an automaton for each process by relying on the notion of Labeled Transition System (LTS). The semantics of the interpreter are defined using the notion of tokens. Tokens are used in the automaton to mark the active places of the process. The location of all the tokens in a given automaton describes a configuration. The structure of the interpreter is developed around three main functions:

- `initialConfigurations` returns the set of all possible configurations for the process.
- `fireableTransitions` returns the collection of all possible transitions that can be fired from a given configuration.
- `fireTransition:` fires a given transition and returns the target configuration reached as a result of firing the transition.

These functions of the interpreter are used for the development of the LTS graph for a given process and the possibility to traverse it using `fireTransition`.

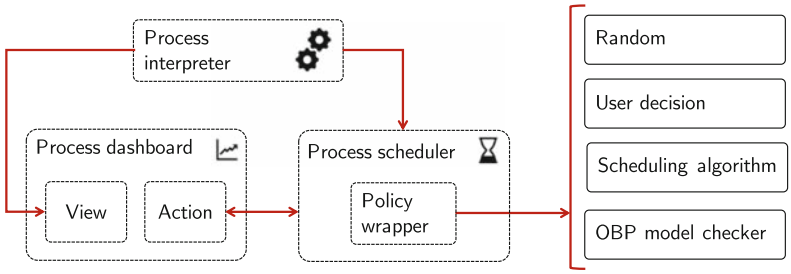

Fig. 3. Process scheduling architecture through policies

The semantics[3] defined in the process interpreter are local to it. Other components use these functions to access the current state of the process model and to see what states of the process model can be attained next. It is important to note here that the process interpreter develops a single LTS graph for all the collaborations and the parallel processes executing in different pools.

Where the process interpreter is responsible for interpreting each individual activity and enacting it, the process scheduler chooses the control flow between these activities. This choice of activities is forwarded to the interpreter, which is then capable of enacting the complete process model. For every non-deterministic choice, the process scheduler chooses a control flow branch according to the policy selected in the policy wrapper. Figure 3 shows the role of policy wrapping for the process scheduler. Under the current implementation, the policy wrapper can choose one of the following policies for scheduling the process:

- *Random:* This policy deals with the non-determinism in the process in a random fashion. It does not evaluate any guard conditions for choosing one of the LTS graph branches and randomly selects any one of them.
- *User decision:* This policy delegates the responsibility of choosing one of the branches to the user. This responsibility is delegated using the Process dashboard. It is used when the system is enacted with a human in the loop.
- *Scheduling algorithm:* This policy allows the use of a user-defined algorithm for choosing between two branches of the LTS graph. Such algorithms can also evaluate the condition expressions associated with the outgoing sequence flows from the decision node by using Decision Model and Notation (DMN) tables. This policy also adds the possibility to use complex user-defined algorithms from decision support systems, artificial intelligence, *etc.*
- *OBP model checker:* This policy is specifically defined for connecting the process interpreter with the model checker through the process scheduler. The objective is to capture all possible control flows of a process without restricting the interpreter to only those triggered by specific input choices. Hence, this policy passes a holistic graph of all possible states to the model checker for formal verification.

[3] The objective of this article is not to present the in-depth semantics of the process model, instead we focus on presenting the global methodology of the framework.

Process Verification: OBP model checker[4] is responsible for process verification in our framework. It relies on the identification of contexts and their exploitation during the exploration of the models. Observer-Based Prover (OBP) uses Context Description Language (CDL) for formally describing the context of the system [6]. It reduces the state space for proving/disproving the correctness of the system with respect to the relevant properties. The principle is to lead the explorer in a way that it does not concentrate its efforts on the exploration of the complete space of the behaviors, but on a relevant restriction of the latter for the verification of specific properties. Once the state space is reduced, it becomes easier to verify the properties. These properties are defined as part of the context description using the CDL language. *OBP model checker* component is connected to the process interpreter through a *process scheduler*, allowing it to the construct and traverse the LTS graph of all the possible configurations for a process model. A configuration for OBP model checker is a unique state of the process, where a subset of activities from the process (including all or none) have been executed. Each activity in the process model is considered as a transition that moves the process from one configuration to another in this directed graph. Using this connection to process interpreter, the OBP model checker component can focus on the formal verification of the temporal properties of a given process.

Considering critical and complex nature of the systems, we focused on the safety properties of the process. The architectural choice of linking the process model to the model checker through the process interpreter and process scheduler is motivated by the second research objective. Traditionally, an *original* model to be verified is rewritten in a chosen formal language. After the formal verification, it becomes hard to verify (and maintain) the equivalence between the model defined in the formal language and the code/application generated from the *original* model. This architectural choice allows us to carry out model checking on the process models based on the semantics defined in the process interpreter. The same process interpreter is later used as the core of the process-centric critical and complex system. Hence, this issue of verifying and maintaining the equivalence does not apply in our framework.

A process dashboard component serves as an interface to manage the enactment of a process model. The execution of the activities is triggered by the interface provided by the process dashboard. The process dashboard is mainly composed of two components: *view* and *action*. The *view* component gives information about the current state of activities in the process model in order to allow process monitoring. This component takes the information about the enacted process from the process interpreter. The *action* component provides an interface to process dashboard for taking user inputs. These actions depend on the type of policy chosen by the process scheduler. For example, in case of the *random* policy, a user can simply choose start, next, stop, pause and resume actions, however the *user decision* policy allows the user to choose specific activities for the control flow.

[4] http://www.obpcdl.org.

Process Simulation: When process models are used to define the behavior in critical and complex systems, they need to be accompanied by methods that can analyze the processes. Model checking techniques analyze the structure and behavior of the process model by focusing on the flow of control. In order to respond to the third objective, we introduced a mechanism to separate the control flow analysis of the process from the analysis of the actions defined in individual activities. In order to analyze individual activities and their impact on the global behavior of the process, we use agent-based simulation and visualization approach. Like traditional formal verification approaches, existing simulation techniques are based on the static description of the process without taking into account the semantics defined in the process interpreter. Our objective is to propose a methodology to remotely simulate the process models according to the semantics defined in the process interpreter. Through remote simulation, the idea is not to orchestrate between multiple simulations (*e.g.* federates in a federation of HLA [1]), rather it is to control the execution of the scenarios, for both simulation and visualization, through the defined process models.

Similar to the approach followed for model checking, we linked the simulator to the process interpreter instead of analyzing the serialized process model. Instead of connecting the process interpreter directly with the process simulator, we developed a *service dispatcher* component that allows to access the functions of the process interpreter remotely. Directly linking a process model to the simulation framework enforces the predefined control flow on the agent-based system. However, putting a process interpreter in between the both, allows autonomous agents to develop localized process behaviors. Then the coordination of these localized process behaviors helps to emerge the global behavior of the complete system. One way of validating the process model is to analyze if the emerged behavior of the agent-based simulation conforms to the process specifications and is aligned with user expectations.

We used an open-source agent-based simulator, DirectSim, for the simulation of process models. The simulator itself is extensible using a plugin mechanism. In our implementation, we developed a plugin for the DirectSim simulator that adds a TCP/IP client to the simulator. This way, the simulator is able to communicate with the process interpreter through the service dispatcher. Once the simulation is launched from DirectSim, the user can control the simulation using the process dashboard. Currently, we are using the *user decision* policy to control the simulation of process model, but in future we plan to develop advanced scheduling algorithms for process simulation.

2.2 Proposed Methodology

In the proposed framework, the architecture for the process-centric system is based around a methodology that seeks early validation of critical and complex systems. The proposed methodology, as shown in Fig. 4, exploits this architecture

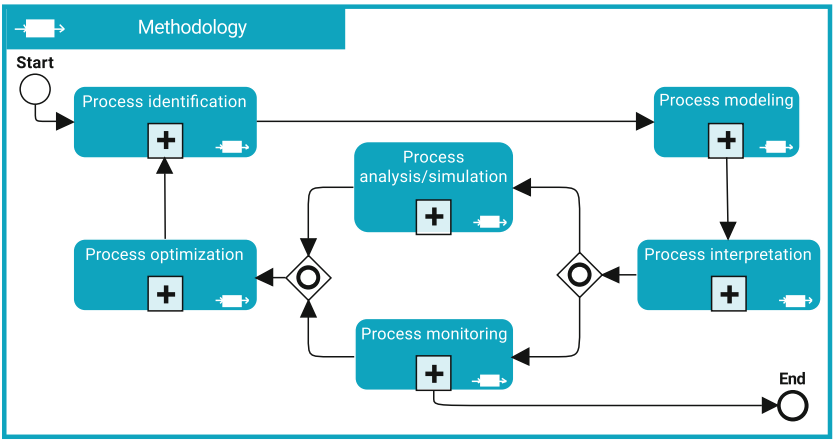

Fig. 4. Process modeling methodology in the proposed framework

to reinforce the objectives outlined in Sect. 1. This methodology involves the following sub-processes:

- *process identification:* Process identification is handled through the *C4: Standard Processes* viewpoint of the NATO architecture framework. This viewpoint also establishes the traceability of the identified processes to the capabilities supporting them. The tools needed for developing a hierarchical table of processes are simple word processors or spreadsheets.
- *process modeling:* This sub-process focuses on modeling the processes identified in the first sub-process. In order to keep Fig. 4 simple, we did not show the iteration between process identification and process modeling. An alternating cycle between process modeling and identification is used for the refinement of processes. Another important activity within this sub-process is the development of multi-layered process model, where the logical activities describe the business process at one level and the resource functions describe the application process at the other level. The tools used for this activity are process modeling editors e.g. we use Mega HOPEX.
- *process interpretation:* This sub-process focuses on the interpretation of the process model in a way that all the activities of a given process can be executed with a well defined flow between them. The interpretation of a process model locks the operational semantics defined for a given process. Both the process model and the operational semantics defined in the interpreter together are responsible for the execution of the process-centric system.
- *process analysis & simulation:* This sub-process aims at the analysis of the process using two methods. Model checking techniques are used for the verification of properties concerning the control flow of the process. The second method is to simulate the process for strategic management, planning, understanding, training and improvement. The simulation of the process model is mostly focused on the analysis of the actions inside individual activities.

– *process monitoring & enactment:* This sub-process is responsible for the control and operational management of the process. After the analysis and simulation of the process, the simulated services for the sub-systems are replaced with the actual system services, to meet our fourth objective. The shift from the simulated services to the actual system services is gradual. New sub-components modified/added to the system, even after the deployment, can be simulated using the proposed framework.
– *process optimization:* Finally, this sub-process ensures a constant process improvement for the specified processes. The recommendations from the process analysis/simulation and process monitoring/enactment serve as an input for optimizing process specifications.

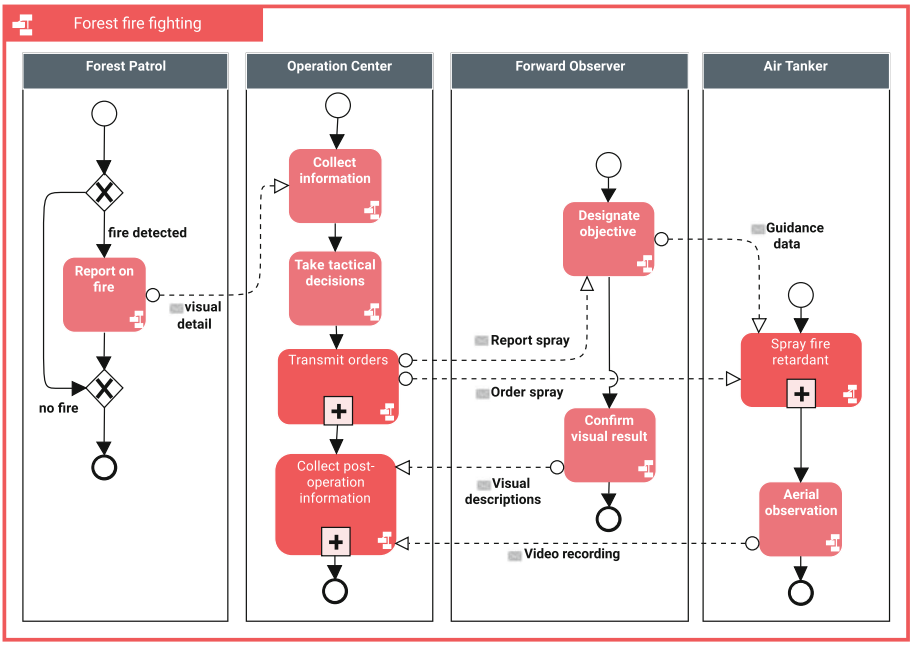

Fig. 5. Application process for the case study

3 Case Study

The methodology proposed in this article has been applied to real-life case-studies in the military context. Bound by the non-disclosure agreement, we are obliged to adapt the actual military case study to secure some sensitive information and processes. This adapted case study, shown in Fig. 5, illustrates the process of launching/spraying fire retardant at a wild fire identified in a forest. Four participants that coordinate to realize this process are air tanker, operation center, forward observer and forest patrol. The forest patrol identifies the

fire and reports it to the operation center, which then transmits the orders to the air tanker. The air tanker then follows the coordinates given by the forward observer and sprays the fire retardant. The forward observer visually observes the results and reports back to the operation center.

Process Development: The *C4: Standard Processes* viewpoint of NATO architecture framework (NAFv4) is used to identify the activities involved in the process. This viewpoint used a tabular representation with a hierarchy of activities with corresponding links to capabilities. Then, the identified activities were modeled as business process model in the *L4: Logical Activities* viewpoint. In our case study, we used the HOPEX tool[5], which supports the development of NAF models. The business process model developed with this tool categorizes the activities in swimlanes and provides the control-flow semantics that are fairly close to that of BPMN Collaboration Diagrams.

Once the business process is defined, it needs to be refined to an application process to get low-level enactable/executable activities. The application process in our case study was captured by the *P4: Resource Functions* viewpoint. The same tool, HOPEX, was used for the development of the application process. The adapted version of this case study, shown in Fig. 5 presents 9 activities/sub-processes (instead of 24 in the actual case study). A mapping between the business and application processes was assured using *L4-P4: Activity to Function Mapping* viewpoint. HOPEX allowed us to serialize the process model in *.bpmn format.

Process Control Flow Analysis: Our process interpreter took the serialized process as input to enact the process according to the defined semantics. Once the process model was loaded into the process interpreter, the process scheduler could access the initial configuration using the `initialConfiguration` interface and traverse the complete model using `fireableTransitions()` and `fireTransition` interfaces. We initially chose the OBP model checker as the scheduling policy, allowing us to continue with the formal verification of the control-flow of the process model. Context models and properties were expressed in a Context Description Language (CDL) so that the model checker, OBP (Observer-Based Prover), can construct its exploration space with all the possible configurations for the given process. It is important to note here that the complete process had four parallel collaborating processes, defined in four different pools. OBP model checker follows the technique of exploration space reduction by focusing only on the configurations that are relevant for the verification of chosen properties. In the process of the adapted case study, OBP reduced the exploration space to 13 states and 12 transitions (considering each collapsed sub-process as single activity). This tool allows one to visualize the state space and analyze the variable values inside each state. Figure 6 depicts a fragment of the exploration space developed by the OBP Explorer.

[5] https://www.mega.com/en/product/hopex.

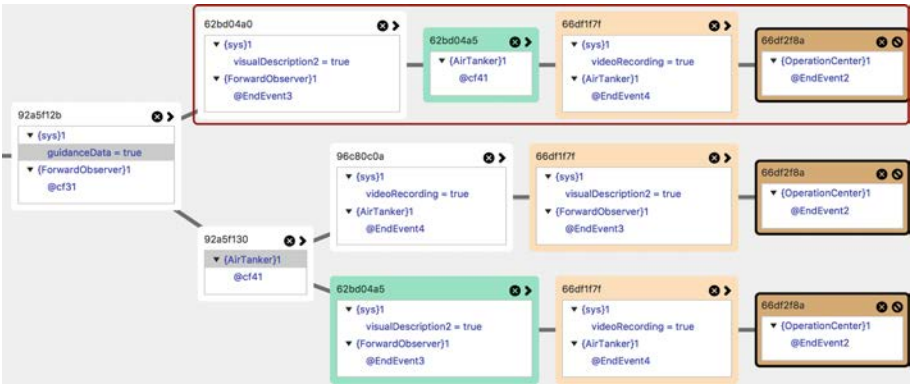

Fig. 6. Fragment of the configurations produced by OBP model checker

The analysis of the adapted case study process model gave insights into the collective behavior of the four collaborating sub-processes. Once the *Designate objective* activity of the forward observer is executed, it sends a message to the *Spray fire retardant* activity of the air tanker with *Guidance data*. Note that the semantics defined for the *send task* in BPMN collaboration are non-blocking *i.e.* once the send task has been completed, this activity can continue its normal execution. Thus, after sending the message, the forward observer can continue with the next activity, *Confirm visual result*. At this point in time, there is a non-determinism between the *Confirm visual result* activity of the forward observer and *Spray fire retardant* activity of the air tanker, as shown in the Fig. 6. In this case, if the former activity is executed before the later (the branch depicted in the red box in Fig. 6), it does not make a logical sense to witness the result before the action is performed. In this case, the formal verification of this process model through our model checker suggests us to add a construct between these activities in a way that the *Confirm visual result* activity is always executed after the *Spray fire retardant* activity has been completed.

Process Simulation: Once the process model has been verified through the model checking activities, one can guarantee that the model carries the tested properties. However, in our case, we had an added guarantee that the model as defined and *the way interpreted by our process interpreter* carries the verified (safety) properties. The next part of the methodology was to connect the process interpreter with the DirectSim Simulator. In order to carry this out, first we changed the policy to *User decision* in the *policy wrapper*. This puts the user in command for controlling the execution of the process. The view of the dashboard changes according to the policy and the user can chose to start and stop individual activities. In this case, whenever a branching in the LTS graph is approached, the system relies on the user input for deciding between the choices.

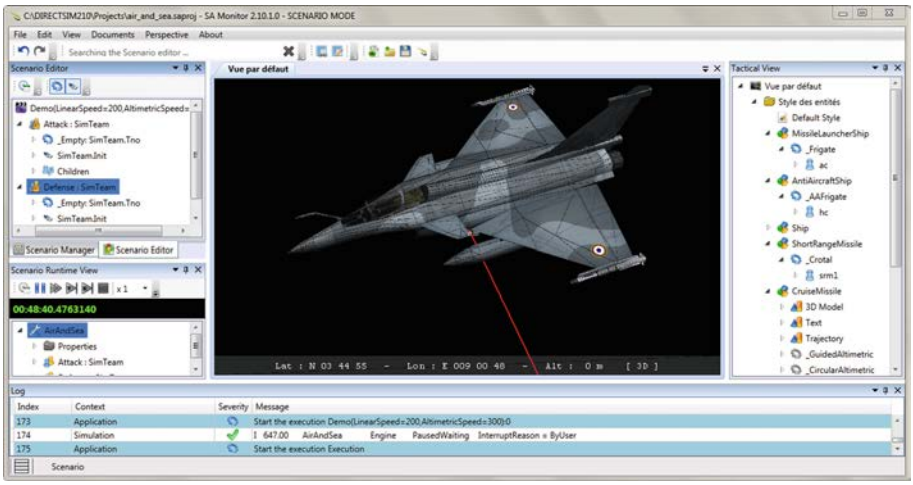

Fig. 7. DirectSim simulation and visualization

For the DirectSim simulation of the process model, we developed the simulation of the process activities using the Visual Studio Editor as suggested by the simulator specification. DirectSim offers a domain specific language for the development of simulations. These simulations are developed around the concepts of agents. In this case study, all the participants of the process were considered as individual agents. DirectSim also serves for the visualization of the 2D/3D simulations. In this case, each of the agents was given a 3D decorator to be used in the simulation. The visualizations developed in DirectSim are multilayered. The background layer was chosen using the latitude/longitude coordinates for the 3D model of the world. The implementation of each individual activity was programmed in the DSML e.g. the flight of the air tanker from the runway to the point of action. Once the simulation was developed in the DSML, it was used to generate the C# code for the simulator. The plugin extension that we have developed for the DirectSim simulator allowed us to control the simulation from the process dashboard. This remote simulation was realized through a TCP connection between the *Service dispatcher* component and the DirectSim plugin.

In this case study, we developed the simulation of the actual system on an agent-based simulator, as shown in Fig. 7. As a process-centric system, the process interpreter was placed at the core of the system to control the execution of the simulation. We did not replace the simulated activities with the actual activities in this adapted case study. However, we can imagine a communication interface with the forest patrol participant of the case study. In this case, when the forest patrol sends a visual detail of the event through the communication interface, the *Report on fire* activity of the process will be consider as executed.

4 Related Work

Some researchers have looked into the possibility of generating a business system simulation model using a set of executable BPMN models [10]. A practical problem with such approaches is that they suppose an exhaustively described process model to a very low level detail of the system under study. Hence such approaches tend to mix the business level processes with the application and operation level processes in a single model. Using such an approach might work for a specific simulation but plagues the business process with a lot of noise that renders it unusable for other strategic activities. For these reasons, we suggest the use of process models with varying level of abstraction, in the context of enterprise modeling, that are mapped together for traceability.

Oliveira & Pereira suggest that after a decision point (gateway) different branches have distinct probabilities of being followed in runtime, so they propose that every branch has to characterized by a probability [8]. We propose the use of policy wrapper in our framework that allows the user to chose between multiple options to deal with non-determinism. For example, the *scheduling algorithm* policy allows us to couple the decision points of a process model with genetic algorithms or machine learning for decision support systems.

Mappings between business processes and agent based simulation have already been proposed for the validation of processes [14] and description of agent-based conceptual models [13]. These approaches focus on the simulation of the process models for their verification. We decomposed process model verification into action verification and process verification, where the former focuses on the verification of individual activities and the later for the verification of the control flow between them. We propose using process simulation for action verification and model checking for the process verification. For the simulation of process models, we chose an architecture that links the process interpretation and simulation through a service dispatcher. Some approaches suggest a distributed simulation of BPMN models using HLA [1] framework [7,9]. Their focus is mostly on dividing a simulation in multiple sub-functions and executing each (federate) on a different (possibly geographically distributed) system [9]. As for now, we did not focus on federating multiple simulating components on different machines, but this remains a perspective for us. We also plan on carrying out experimentation and sharing the results when the original sub-components of the system are replaced with the simulated functions of the process.

5 Conclusion

We present a framework with architecture, methodology and associated tools for the development of processes in the context of process-centric critical and complex systems. The proposed architecture uses the business and application level process models from NATO architecture framework and interprets them for model checking, simulation and process monitoring. The framework keeps the process interpreter at its core, where the scheduling approach for the activities is

chosen by a policy. These scheduling policies allow using the same interpreter for model checking, simulation and the final system implementation. Model checking is performed using an observer-based prover that is linked directly to the process interpreter. Instead of taking a static serialized process model as input, it takes the dynamic interpretation of the process model as input. We present an approach for simulating the activities of a process model using process dashboard to control the simulation. Our approach is explained through an adapted case study of a real life military project.

Acknowledgment. We thank French ministry of the armed forces and specifically the Directorate General of Armaments for funding this research work.

References

1. IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) - Framework and Rules. IEEE Std. 1516-2010, August 2010
2. ArchiMate 3.0.1 Specification. Open Group Standard, August 2017
3. NATO Architecture Framework version 4.0. NATO Std. NAFv4, January 2018
4. Cornu, F., Garnier, A., Audoly, C.: Simulation of the efficiency of a system of drones in a mine warfare scenario. In: Undersea Defence Technology, pp. 1–8 (2008)
5. Corradini, F., Fornari, F., Polini, A., Re, B., Tiezzi, F.: A formal approach to modeling and verification of business process collaborations. Sci. Comput. Program. **166**, 35–70 (2018)
6. Dhaussy, P., Boniol, F., Roger, J.C., Leroux, L.: Improving model checking with context modelling. Adv. Softw. Eng. **2012**, 9 (2012)
7. Falcone, A., Garro, A., D'Ambrogio, A., Giglio, A.: Engineering systems by combining BPMN and HLA-based distributed simulation. In: 2017 IEEE International Systems Engineering Symposium (ISSE), pp. 1–6. IEEE (2017)
8. Freitas, A.P., Pereira, J.L.M.: Process simulation support in BPM tools: the case of BPMN. In: 5th International Conference on Business Sustainability: Management, Technology and Learning for Individuals, Organisations and Society in Turbulent Environments (BS 2015) (2015)
9. Gorecki, S., Bouanan, Y., Zacharewicz, G., Perry, N.: BPMN modeling for HLA based simulation and visualization. In: Proceedings of the Model-driven Approaches for Simulation Engineering Symposium, p. 11. Society for Computer Simulation International (2018)
10. Guizzardi, G., Wagner, G.: Can BPMN be used for making simulation models? In: Barjis, J., Eldabi, T., Gupta, A. (eds.) EOMAS 2011. LNBIP, vol. 88, pp. 100–115. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-24175-8_8
11. Holzmann, G.J.: The model checker SPIN. IEEE Trans. Softw. Eng. **23**(5), 279–295 (1997)
12. Inselberg, A.: Multidimensional detective. In: Proceedings of the 1997 IEEE Symposium on Information Visualization. IEEE Computer Society (1997)
13. Onggo, B.S.: Agent-based simulation model representation using BPMN. In: Formal Languages for Computer Simulation: Transdisciplinary Models and Applications, pp. 378–400. IGI Global (2014)

14. Pascalau, E., Giurca, A., Wagner, G.: Validating auction business processes using agent-based simulations. BPSC **147**, 95–109 (2009)
15. Sheth, A.: From contemporary workflow process automation to adaptive and dynamic work activity coordination and collaboration. In: International Conference of Database and Expert Systems Applications, DEXA 1997, pp. 24–27. IEEE (1997)