Response letter to EMISAJ Manuscript entitled "Multi-level modeling with Openflexo/FML- A contribution to the MULTI process challenge"

September 7, 2021

Dear Editor,

Thanks for your message of August 16th, 2021, in which you suggested preparing a revision of our manuscript **Multi-level modeling with Openflexo/FML-A contribution to the MULTI process challenge**.

We are grateful to the reviewers for their interest, effort, and suggestions. We detail in the following pages how we have addressed the reviewers' concerns; we are confident that the paper is now improved, and hope that you will agree. We have organized the responses to the comments on a per-reviewer basis, focusing, as suggested, on:

- Improving the presentation.
- Provide a more explicit description of how the multi-level modelling emulation is achieved.
- Discussing the general advantages and limitations of the multi-level emulation, including a clarification of what the implications would be of supporting more "levels" than the challenge required.

Moreover, the paper has been thoroughly revised to fix minor issues and to give the additional clarifications required by the reviewers. Please let us know if you need any further information or clarification.

Sincerely yours,

Sylvain, Guillaume, Caine, Joel, Jean-Christophe, Salvador, Fabien and Antoine

Comments by Editor

> the reviewers have made suggestions for improving the
> presentation but also raised a number of issues requiring
> clarification. An overarching theme is the request for a
> more explicit description of how the multi-level modelling
> emulation is achieved and what its general advantages and
> limitations are, including a clarification of what the
> implications would be of supporting more "levels" than the
> challenge required.

TODO.

> We very much welcome a "conventional" response to the
> challenge but would like to see the respective reviewer
> questions addressed in a revised version which is explicit
> as possible about the advantages and the disadvantages of
> the emulation. You may find the references listed below
> useful to keep explanations concise by referencing related
> work (note that reviewer 3's \Type-Instance pattern" is
> also known as the \Type Object" pattern). You should also
> respond to other concerns reviewers raise, either by making
> changes to the submission or in the form of an authors'
> response.

Je ne sais pas si c'est conventionnel ? Fig 3 ?

TODO. We may have to create a new section.

> Finally, to achieve consistency with other submissions,
> could you please change the subtitle to "A Contribution to
> the Multi-Level Process Challenge"?

Of course we can. DONE.

Comments by Reviewer #1

> In section 2 Technology the authors describe the foundations of the Openflexo approach. Here it would greatly help the reader to see an example how the approach works for traditional modeling languages. For example, how would you realize a small subset of BPMN with the approach? This could briefly be illustrated so as to give the reader an immediate understanding. Especially, the 'metamodel'-agnostic approach seems very interesting but I would like to see how that is used for traditional modeling approaches.

TODO.

> Also, at the stage of section 2, the characterization of the FML language should be made clearer. The metamodel in Figure 2 seems mainly conceptual - but at the same time you refer to it as a basis for implementation. Could that be clarified? eg you have the class "Behavior" - which is a concept and I did not understand right away what it is composed of. On the other hand you have for example the class "ModelSlot" which seems like a very technical realization. Also, you note that roles have types but I could not find any types in the metamodel (?). Thus, I propose to include a more detailed version of your metamodel at this stage. Please note that EMISAJ does not have a page limitation, so no worries if that should require more space.

TODO.

Completer 2 Technology et donner un petit exemple. (Ne pas parler d'histoire...)

Connecteur EMF ; on se connecte à un MM BPMN (v1 - reuse) et (v2 - clone) on métamodélise BPMN en choisissant les concepts qu'on garde (comme tout le monde). Ca on le range dans un VM qu'on peut réutiliser. Conformance de v2 par consensus.

/EMF ... generation de code ; /Federation interpretation exemple simple

Prendre un micro-BPMN en exemple. Process-Tache ?

MF2 à simplifier.

interpréteur de modèles FML

Behavior —*> Instructions

FML interprété ; tout n'est pas dans le MM structurel. Label Fig2 à revoir. (Abstract MM)

Expliciter le choix des notations. Proche de UML mais pas UML.

Légende VM + FlexoConcepts + …

Fig 2… ontology
Fig 4… légende explicite dans le texte : VM + concepts

> In Figure 5, the base metamodel for representing processes is shown - however it is not linked to the concepts you introduced before. Would it be possible to maybe graphically highlight how it relates to the metamodel in Figure 2? Besides, as multi-level modeling is much different to traditional modeling, you may want to consider using a different notation rather than UML class diagrams - as these are commonly known for traditional modeling approaches. When reading the paper, I actually marked the references to types in yellow in my copy - ie from ProcessType to TaskType and from ActorType to ActorType (on the attribute levels) - this helped me to see more clearly where you have references from attributes to types - but that's just my way of doing it.

Fig3 dans le texte expliciter les relations entre VM.

Colored UML (Type≠ Instance)

TODO.

> Regarding Figure 6 and Figure 7: In Figure 7 you show the process incl. sequence connectors - however in Figure 6 these seem to be missing - maybe to reduce complexity? I would suggest to probably focus on a subset of the process from Figure 7 and also show how the connectors are realized. To me this would increase the comprehensibility / consistency. Ie I could then infer exactly, how elements from the concrete model have been realized in the background.

Mettre en cohérence fig6 et 7 (mêmes concepts présents ; ex. Gateway and subconcepts) => add Gateway

TODO.

> in Figure 9 you show the model editor. Could you please also add a screenshot of the metamodel editor?

add screenshot

TODO.

> There are some minor typos throughout the manuscript (eg also regarding the reference Jeusfeld2019; choose -> chose; axis -> axes; aN FML virtual model...; etc.)

TODO. choose → chose OK ; axis → axes OK ; je ne comprends pas pour Jeusfeld2019 (deeptelos2019) ni pour aN FML (je dirais "a FML", mais je puis me tromper)

an FML ? ask C. Cao

Comments by Reviewer #2

Déplace (étend) le problème classification des approches MLM au process et organisation pour répondre au problèmes multi-niveaux.

> The basic idea of model federation is interesting, its primary focus is not on creating multi-level models. However, as the approach is quite flexible, multi-level behavior can be emulated by model federation. This fact is possibly the most serious advantage and disadvantage of the paper at the same time. Advantage, since it brings us a completely new, different way of thinking, and disadvantage since it does not really help in categorizing/unifying multi-level methods and { for me { it also feels a little bit off-topic in this special issue. At this point, it should be mentioned that the authors managed to cover/solve all requirements of the challenge, thus the paper, as a solution to the challenge fits well.

Our MLM approach is not embedded in a language => we use a process that follows the history of the problem/challenge evolution and build a « hierachy » of models.

tailored MLM ?

TODO.

Answering this is going to be tricky!

> The paper gives us a short introduction (somewhat overlapping with the abstract, even at the level of sentences) and then the technology is detailed. The section is a bit hard to understand at first, the difference and the relationship between federated and virtual models could be explained in more detail especially considered that the challenge solution uses virtual models only. It is not clear, how federation works in this case.

Voir #Reviewer1

TODO.

> The third section contains an overview of the solution. Here, virtual models are instantiated to virtual model instances conforming to the virtual models acting as the metamodel for the virtual model instances. The question naturally arises: how does it work, if we have more abstraction levels? In this section, the authors also discuss linguistic and ontologic instantiation. The explanation refers to Section 4 resulting in a hardly understandable analysis of different instantiation methods. Figure 4. does not help in clarifying the concepts either (e.g. using the name InstanceInstance seems to be a very bad decision). From the point of the muli-level behavior of view, the interpretation of the instantiation relationship is the most crucial one that is why this section must be clarified. Where are the rules of ontological instantiation defined, how are they enforced? The text mentions the (linguistic) instantiation mechanism of the FML Language... how does it work exactly?

comment on ajoute un niveau ? on ajoute un VM (dire explicitement avec Fig3)

Voir #Reviewer1

TODO.

> Section 4 elaborates the solution in detail. Figure 5 presents an overview of the model elements handling P1 to P19. The diagram uses a UML-like annotation, which helps in making it understandable, but it also has a strange decision: roles (attributes) having a modeled concept as their type are visualized by links, but instead of using rolenames on the links, the concepts (classes) contain them. I strongly suggest following the UML guidelines here and use rolenames (similarly to multiplicity).

nom de rôle = attribut, sur la bonne ligne ! Dit dans le texte

TODO.

> Overall, the solutions for the requirements are correctly presented in this section, but there are some spots not clear enough. E. g. regarding ontological instantiation, the solution uses two kinds: single and multiple. More details would be welcome to see, what their exact semantics are and where do we enforce these semantic rules (is it enforced by the engine? Is the semantic hard-coded?). What are the contexts of the constraints (to which concept are they attached to)?

Revenir sur FML et l'interprétation. Il y a du code non présenté.

sur les concepts nécessaires (méta ou non)

Le behavior est défini dans les constructeurs des concepts Instance et MultiInstance (MultiTypes ?)

TODO.

Expliqué en partie dans le 5

éviter d'utiliser le mot contrainte

Instance/MultiInstance est un choix de conception ; on aurait pu avoir MultiInstance seulement et ajouter des contraintes dans les niveaux du dessous…

Pas contraintes FML ; plutot des préconditions

Expliquer dans le texte... plusieurs utilisateurs. On définit la sémantique en proposant des actions (avec une sémantique) pour manipuler les modéles, méta et instances

businessAction = transformation + usage

> In Section 4.1.2 a behavior called newProcess is defined. Is it called automatically (by the framework) when creating a new process, or should the users call it? Is this a convention, or are such methods configured somehow to replace the usual constructors? According to the text, \same pattern applies" in several cases, but these patterns could be also included, because it is not completely clear to what part of the behavior \same" refers. In same cases, the diagram uses \/type", while in the code it is \type" { why? The specialization of Actors is encoded by FML, wouldn't be easier to use the built-in specialization instead?

same pattern of instantiation?

TODO.

cf Reviewer1

> In Section 4.2 it is mentioned, where the solution uses linguistic instantiation, but it is not clear by which component it is handled and what rules are followed when instantiating the concepts. In Section 4.3, tooling is elaborated. I suggest removing redundant information about the two editors and add more details on how the concrete syntax (e.g. the texts in the box of Fig. 9.) is defined.

The tool precedes the language!

TODO.

> In Section 5, the meaning of the category Conceptualization is hard to understand, does this mean that a concept is created that covers the requirement completely? In the case of ontological instantiation, the term \dual concept nature" and \instance nature" is mentioned... what do these terms mean?

DONE : check p13

Ontological instantiation

reformuler p13

TODO.

> Section 7 contains a rather short elaboration on related approaches. Although I agree that it is enough to focus on previous solutions here, much more detail and a comparison would be required.

TODO.

> As Section 6 shows, the authors know exactly, why their solution may have difficulties as a multi-level modeling approach despite solving the given challenge. The solution is basically a smart, two-level modeling solution of the MULTI challenge. Multi-level behavior is emulated by constraints and additional mechanisms particularly written to this challenge. The result is a working workbench fulfilling the requirements but having limited reusability and extensibility. For example, it would be hard to increase the number of abstraction levels as class-like and object-like concepts are handled differently. Another problem may arise if we would use the framework as a prototyping tool and change the metamodel and the models at the same time.

???

Classical Multilevel approaches are a choice of constraints. Evolutions are easy if rules are consistent. If they were different, evolutions would be much harder too.

TODO.

> Minor issues/questions:

- Are there any tools supporting model federation besides Openflexo?
- Section 2. "In contrast to approaches that compose metamodels into a single large metamodel
- grouping all needed entities,..." - I do not see, why this is relevant here
- Figure 5 – concept Sequencing – the roles "in" and "out" are not represented by lines... why?
- What does the term "sub-concept" mean exactly? Please explain it before the first use
- ProgrammingLanguage concept is an enumeration... how is it modelled?
- "The Acme metamodel also defines a specialized task type Coding" – isn't it CodingTask?
- "The value of the property producedArtifactTypes" – is it property, or role?

TODO.

The idea is to group all the answers to the minor questions here.

Comments by Reviewer #3

> (1) The Type-Instance pattern is key to the authors approach to the challenge. However, the benefits of this pattern are not explored in the context of MLM. For example, the authors do not really explain approaches such as Clabjects or potency levels as they compare to the Type-Instance pattern (or the challenge). Why it the pattern better? Is it just a different approach? What can it do that other approaches cannot? Perhaps the benefit is that the Type-Instance pattern can be implemented in most modelling and implementation technologies whereas other approaches need special facilities? Finally, in terms of the pattern, why should anyone care - why would should they use the pattern?

See last comment R#2

TODO.

> (2) In section 1 the authors list 2 key reasons why a strict approach to modelling is sub-optimal. However, they do not establish the case that their approach uses these 2 features to any concrete benefit. It would be appropriate (perhaps at the end of the article) to revisit these features and to clearly demonstrate that their approach supports them and that they have concrete utility in some way in the context of the challenge.

Il faudrait qu'on imagine une évolution difficile pour les MLM.

à discuter

TODO.

> (3) One of the key benefits of the Type-Instance pattern is that it provides a 'hook' on which semantics can be defined using constraints. There is a reified relationships between a type and its instance and therefore conditions can be defined that establish both static and dynamic semantics. I am confused regaring why this is not a key feature of the semantics for the challenge as it is implemented by the authors. Constraints are provided very sparingly compared to implementation of behaviour in FML. Unfortunately, since the dynamic semantics of FML is not provided (and is bespoke) this is less clear than it might otherwise be for the reader. Perhaps it would be better to use constraints in the context of the Type-Instance pattern to specify semantics? It would be more abstract than the current 'method' style semantics.

TODO.

> (4) Section 6 lists a number of benefits of the approach.
> Unfortunately, the presentation in the section is very
> general and none of the claims are justified in terms of
> concrete descriptions and/or examples. This section seems
> to be very important and could be of great interest to the
> reader. Perhaps it would be appropriate to reduce some
> of the rather implementation heavy descriptions earlier in
> the paper and increase the analysis in this section? In
> addition section 4.3 notes that MLM is key to the ability
> of Openflexo to create tooling. Again, this is very
> interesting, but no concrete details are provided as to
> how Openflexo uses the level agnotic features of the models
> to create tools. Can the essence of this be captured for
> Openflexo and then described in terms of how it facilitiates
> the tooling arising from the challenge?

TODO.

> (5) The paper makes an issue of the model federation
> approach supported by Openflexo, including the use of
> model slots. However, after introducing these concepts
> they do not seem to be important to the rest of the
> article. Perhaps this is an implementation concern that
> could be elided as its inclusion does not seem key to the
> contributions of the article.

TODO.

> Some minor issues:
> (a) In fig 5: why are the in/out gateway features shown
> as properties and not as associations?
> (b) On p. 7 'specialization' -> 'specializations'.

TODO.