Transcript - Group 3 "Modelers", Interview 1

I … Interviewer (BLINDED)
B … Expert
(Unv.)… Incomprehensible passage
(...) … Pause longer than 3 sec.
(   ) … Comment
// …// … Speaker overlap

Transcript

1. I: Okay, recording started. (...) Okay, hello and thank you for taking the time to do this interview with me. [redacted]. Would you please perhaps like to briefly introduce yourself and explain your connection to BPMN or to process engineering, manufacturing engineering, process modeling? I would like to ask you not to state your name in the process, but only the following information. Starting with your job title and description of employer. Basis of expertise on the research topic, education or professional background, and work experience. #00:00:52-1#

2. B: [redacted]. #00:02:44-6#

3. I: Okay, thank you. So a quick introduction to my topic. Our research focuses on developing a methodology to represent continuous processes in BPMN and make them executable in a workflow engine. For this task we have been working on BPMN extensions for continuous processes. Why continuous processes? Because discrete processes have already been addressed in other research and do not present the same difficulties in being represented correctly, modeled correctly, using BPMN. BPMN is already a widely used standard in business process management, and has also found its way into manufacturing. Discrete manufacturing processes can already be modeled using BPMN 2.0. Basically, we want to introduce a methodology to represent such processes in such a way, that they can be understood by anyone in a company, from engineers to managers. This could be achieved by using this notation. Another benefit is also that there are already a number of workflow engines. These are applications that allow these process models to be executed based on the logic implemented for each symbol. We are working with a web based application that is extensible and has implemented several communication interfaces. Therefore, another advantage is interoperability in this context compared to other proprietary, rigid software applications. We want to find out if this technique can also be used to implement digital images (models). Since digital images (models) are used to represent a physical system or process in digital form, usually using data or mathematical models, we had to find a way to represent the flow of continuous processes as they are known from the process industry. For this reason we focused on modeling control loops. The process models should be easy to understand by people with different backgrounds using BPMN. The interviews are conducted to find out how process and control engineering and techniques from business process modeling can be combined and how initial results are perceived by experts like you. We also want to find out, if there are any weaknesses identified by experts and how we can address them. So some terms we will be using. Digital twin. There are different methods to simulate things from the real world, for example real machines. Partly, however, one notices

that it would take more parameters than with normal simulation methods to completely represent a machine in the same way it behaves in reality. A digital twin tries to get as close as possible to the real behavior of a machine or other objects. This should lead to the fact that if something is triggered with a real machine, the digital twin shows the same or a similar behavior. What do we define as continuous processes? I would like to explain continuous processes with process examples. If you take beer brewing here. There we have two possibilities, the discrete non-continuous variant would be if you put the ingredients into a closed vessel, a cattle, for example, ten liters of water etc and simply let the brewing process proceed step by step. In the end a limited amount of beer will come out. The other continuous variant would be when you don't have a completely closed vessel, but an interconnected chain of vessels where ingredients are added again and again and the beer is taken out again and again. This goes on all the time, continuously, so you can't trace which liter of water goes with which liter of beer. A partial process takes place in the first vessel, while at the same time, the last process step takes place in the last vessel before the beer is finished. And finally, closed loop systems. A closed loop is that kind of logic in the form of hardware or software that makes continuous brewing possible. When you have a process like continuous brewing you have to look at how to avoid bad beer, of course, while the process is running. You want to keep the quality at a certain point. With incremental brewing you only have the ten liters where something can go wrong and then you do it better with better parameters with the next ten liters. But what if you run the brewing process continuously all the time and the beer is being produced constantly, continuously? Then you have to check the process and see that you get the right quality of the beer. This means, testing on measuring values that describe the quality, checking how these values differ from optimal values, and reacting accordingly. If something is wrong with the sugar or for example also the alcohol ratio, you need to adjust the mixing ratio as well. This means that in a closed control loop, certain values are checked, while the process is running. These are compared with the optimum values and depending on the deviation, the system reacts accordingly. (...) Okay, now we come to the first questions. From a computer science perspective, continuous processes consist of a constantly repeating sequence of state queries, measurements, and regulations. In each case, state queries and regulations are traditional pieces of code that refer to sensors or actuators. In order to consistently formally describe model and subsequently execute such continuous processes, we have identified the following characteristics. I would ask you to rate the following characteristics as important or unimportant, and I'd like to kind of explain your answer also in this context. The first characteristic would be. Different state queries and regulation combinations are independent and may run in parallel. Would you consider this as important or unimportant and why? #00:09:37-2#

4. B: I would say I/ probably yes. Actually, could you restate the meaning of S and R, just, if you can re/ Yes, what do you mean for state queries and regulations? #00:10:03-0#

5. I: State //B: Okay. // queries. We would consider state queries as measurements to get the actual state of a system. To know the actual ratio of sugar, for example, or alcohol in the beer which is con/ #00:10:19-2#

6. B: Okay. Okay then I/ #00:10:21-7#

7. I: /Which is being produced at the time and we/ #00:10:23-6#

8. B: And, so the idea is/ So let's say the characteristic will be the possibility to run those kind of queries and regulation independently one from the other, or to run them in parallel. Is is this the characteristic that referring to? #00:10:47-0#

9. I: It's not like an either-or characteristic. It's like all-including characteristics, so these are independent and they may run in parallel. If there is are defined this way, yeah. #00:11:01-4#

10. B: Yeah. So I think it's important that you could run independently or in parallel those kind of queries and regulation. Because if you have several things that you should monitor and regulate together then that makes sense. Yes. #00:11:20-3#

11. I: Thank you. Regulations always follow state queries. So, we would only consider applying regulations if we know the actual state of the system. Would you say this is important or unimportant? #00:11:37-8#

12. B: I mean. Not/ I mean, I don't think it should be done all in this way, so I mean it/ I mean, what if you have some default regulation, that could apply every time, or? Or yeah, it's like only/ I mean I got the point that you/ it's like apply regulation after you have a clear understanding of what is going on. But then maybe there may be some regulation coming from outside of the environment. That could be, but/ I mean it, it's important to have the possibility to regulate let's say following the state queries. Maybe I wouldn't say that it is always the case. #00:12:50-7#

13. I: So it's like a neither-nor answer from your side, because you could think of process examples, where you have kind of a trigger from outside of the process? #00:13:04-8#

14. B: Yeah, // I: Okay. // that's/ yes so. I mean I would say yes, but I'm more concerned about the always, the term, the always term that is used there. #00:13:16-1#

15. I: Okay. The third characteristic would be the duration of each state query and regulation combination is limited. So we can say, that the/ you get the state of a system and then the following regulation or the respective answer could only be applied in a certain time frame, for example. #00:13:44-0#

16. B: Yes, I mean that makes sense. Because, if you apply that regulation, that action, if you trigger an action too late, it may be that the state has changed. So it depends. It's like time to respond. It's kind of time to respond that you have. So I think it's important to have a sort of temporal limits on the action that you could perform. Since this state could have changed. That probably depends on the time constraint that you have in that scenario. //I: The/ // That's/ Yes, so it's important. #00:14:36-9#

17. I: Sorry, I didn't want to interrupt you. Thank you. The fourth characteristic would be, if state queries deliver certain results, the system is shut down. This should/ How should I explain it? So for/ We kind of introduced the requirement for continuous processes to have certain conditions, based on state queries, to trigger the shutdown of a continuous process, to kind of stop the continuity and go maybe into a discrete process flow. Or into another continuous process, but with another set of parameters. Another set of conditions, for example. #00:15:38-9#

18. B: Okay, so you're saying basically if there/ there should be a way to shut down the process on/ based on some conditions. And yes, I would say 'yes', but I guess, I mean, I imagine it to be not the default scenario, because since it is a continuous process, it's something that you would like to avoid, right? But, probably in cases like maintenance or when something is damaged and should be replaced, then yes, there must be some way to stop the process or to mitigate, let's say, the fact that it should be stopped. So. Yeah, it's/ Yeah, I would say it's important, yes, to foresee some mechanism that allows to shut down the process in a controlled way. #00:16:42-5#

19. I: And prior to a shutdown, the system needs to be transitioned into a consistent state. So, this shall imply that you have the possibility to define kind of a shutdown routine like a/ some process steps or some tasks to be performed prior to the final shutdown of the process. #00:17:10-0#

20. B: I think, yeah, yes. I think those kind of, let's see/ You would like to reach a consistent state before actually shutting down the system. It's like something that probably anyone would like to have, but I can imagine that sometimes foreseeing anything and reaching a consistent state will not be possible. But for sure, it is important to have this kind of mechanism. If there is the possibility to go into a consistent state before shutting down the system, that is what probably anyone would want. #00:18:07-1#

21. I: Okay, and the last characteristic would be, the resulting system must be understandable for people. Or for the human user, you could say. The modeler. #00:18:24-1#

22. B: When you speak about understandability, you speak about the model of the system? #00:18:29-5#

23. I: Yeah, exactly. Yeah. Like the visual representation of the system. #00:18:34-1#

24. B: Yes. I would say 'yes', but also it depends on/ with the target of those models. For instance, if it is used for, let's say, for communication purposes between different actors, so like, I don't know, the business guy and the computer scientist or the engineer. Then, yes. I mean, I understand it's a really important if you/ I mean, it is always important, even if you are using the model for actually executing the process or to give it as input to a machine. It will be less important in the sense that the machine doesn't care much about the understandability of that model. But then when you will have to modify it, to go back to it and to, let's say, to visualize it as an engineer or as a developer. Yeah, then it's important. Also in that case. #00:19:59-3#

25. I: Okay, so following this list of characteristics, could you name any graphical features that you find important for modeling continuous processes? And do they perhaps result in features or characteristics that we have forgotten in our listing? #00:20:23-1#

26. B: Can you go just/ please scroll back to just to see/ So, some other characteristic apart from those, that we have mentioned, that we should include in the model, right? #00:20:42-6#

27. I: If you could think of any example. Anything that comes to your mind. #00:20:47-5#

28. B: (...) Like time constraints. Something like that. It could be incorporated into the model.Or not so, like the fact that, I mean, that/ From a BPM perspective I would model continuous processes like with loops. So kind of a possibility to represent those. Then like to accept the possibility to handle exceptions. That will be great. No, that/ Nothing else comes to my mind, but/ (laughs) #00:21:45-2#

29. I: Okay, thank you. That's great. Exceptions. We have thought about introducing exceptions via the, kind of the fourth characteristic, like state queries with certain results. This kind of state queries would also include, like checking if we have some kind of exception being triggered in some kind of way. So this/ we would consider it to be kind of included here in the first characteristic // B: The fourth characteristic. // fourth characteristic. //B: Okay, yes. // and maybe also the/ You've initially mentioned the time constraints. This would kind of be included with the third characteristic //B: Okay. The duration. Okay.//, the duration. But it's like, yeah. It's good feedback. Thank you very much and it kind of proves our point in this context. And and the third question, before we go to the BPMN extensions would be, if in your opinion you could say, are there challenges in modeling continuous processes? #00:23:08-0#

30. B: There are challenges for sure. (laughs) But/ #00:23:15-7#

31. I: Only if you can think about something that comes to your mind right now. #00:23:19-3#

32. B: So, I mean, even related to what I was saying before. The fact that/ If you have continuous processes, probably you would like that it never stops. So you will like to foresee any possible situation that may occur. So maybe that can be tricky to do. And, also, if you, let's say, if we do not think about any possibility to adapt to the context. I mean, we could think about some degree of freedom for adaptation, maybe? Respect to having every possible exception specified. So because it's going to be tricky, I think. Especially since continuous processes are relying on data. Data can vary. Sensor twitters can produce wrong data sometimes so. I think it's pretty difficult to foresee all the situations that may occur. And, also the fact that, let's say, things should happen on time. Because when you have continuous processes, you could have some,

let's say, chemical reaction or something similar. So being sure that things happen at the right time, may be challenging. (...) I think that's it. (laughs) #00:25:12-0#

33. I: Okay, great, thank you. Okay, coming to the introduction, to the extensions. I will show you processes modeled with BPMN 2.0 also using our extensions. The extensions are intended to provide predefined modeling conventions for routines commonly used in process and control engineering and to help visualize the differences between parallel paths in the process models. The processes are modeled in the [redacted], also known as [redacted]. For a better comprehension of the [redacted] models, I would like to explain to you three additional symbols. We first have service calls, which are basically HTTP requests. You could imagine them only kind of working with Get, Put, Post, the standard HTTP requests, which are sent out. You could also send arguments if you want to, and, yeah, like kind of process also the answers, the outputs. Then we have a script. That script is basically kind of a code snippet which can be implemented in a process flow and you can use the variables defined in the [redacted] also to kind of execute this short script and also introduce, like I've said before, the results of prior HTTP requests and work also with them and process them in a defined way if you want. You could use this for example for short mathematical operations or also for a little bit more complex things. The scripts are basically coded using Ruby. So it's all/ // B: Okay.// Yeah. And the combination of both would be service calls with script. And here you have the exact kind of layout for using HTTP call and using the result you get back for a follow up script, if you want to. So, and now to the extensions of this thesis. We begin with the introduction of the Closed Loop Subsystem Gateway. The gateway is basically a combination of an inclusive and an event-based gateway. It contains branches or edges that are triggered for the state polling and our state queries and regulation phases of the cycle, as well as branches that are executed when abort events are received, cancellation events, basically. The events and tasks in each edge are independent of each other. Thus we fulfill the first of the above characteristics, the above features, and, that individual progressions are independent of each other and they are executed in parallel. The gateway also allows defining the interval duration of each cycle as well as exceedance conditions, wait, cancel, and execution order for state queries and regulations, or which could also be called measurements and control tasks. You can see where to define the interval duration overrun, either cancel or wait, and the measure control cycle execution, either parallel or sequential. So what do they mean? If wait is selected, the next iteration starts when all branches are finished and the specified interval duration is reached. When cancel selected, the interval duration defines exactly the time in which each branch is to be terminated. If the tasks in a branch finish faster, the branch will wait. If not all tasks are finished yet, they will be canceled. Parallel and sequential. With parallel the tasks are executed in parallel according to measure and control events. With sequential, the tasks after control events are executed only after all tasks after measure events have been completed. Kind of the definition of the basic process flow, which follows which task. Intermediate catching events. In a closed loop subsystem specific events are expected to fall into one of the following three categories. Events for state queries or also known as measurements. Events for regulations. Events for the interruption of the closed loop subsystem. There is at least one edge for each event category that originates from the gateway. The edges indicate which tasks are running side by side. As soon as these events occur, the tasks, that are arranged in the edges after them, are also executed. Here is a picture of a closed loop system, in which only events of the three categories are modeled without any tasks following them. So we only have the three events which are not followed by any specified tasks. So if you kind of instantiate an empty system, an empty closed loop system it would look like this. The three event categories which we want to define, are as follows. We first have the measure events receiving events to perform measuring cycles or state queries. The control events receiving events to perform control cycles and the cancel events, receiving events to abort closed loop systems. These symbols indicate the purpose of the following tasks, as stated before. These tasks are only executed when the events are triggered. This means that the measurement event indicates that the subsequent symbols only indicate measurement processes or state queries. The same applies to regulations, control, and to abort events. For state queries and regulations, measurement and control events, we can define a cycle time. This allows us to define the duration of adjustments, regulations, in the system. Depending on whether the closed loop subsystem follows a parallel or sequential, or a wait or cancel approach, the execution runs

differently. These conditions can be used to define the extent to which adjustments are made to the system. Here you can see a closed loop subsystem with a task for a measurement or a state query. In this case, the event for the measurement is triggered every ten seconds. After that, the value V1 is fetched or measured. Wait means here that the new cycle starts only when the measurement is done, i.e. the process in this edge is finished. With cancel, the new cycle started automatically after ten seconds. So here you have the measure event. After it's being triggered, you would go over the task Get process value V1. So here for example, you could send an HTTP Get request to get this exact process value. And here you can/ if you want you can define the interval frequency in hertz. The values expected to change could also be identified or defined for measure events. So to state if you want to use like a global variable which is listed in your variable listings, you can say, 'Okay, this is the exact value that is going to be changed or overwritten' And if you want to also to query more states or measure more values, you can list them and add them to the list here if you want to by using the button add value of course. So in measure, besides the frequency of the measurement event, it is possible to define the value that changes during the process. Control events can also be used to define which controller model is used. So for example, PID, PI, PD, et cetera. These controllers are represented in their mathematical form. The tasks for them are basically calculations represented in fixed sub-processes. In our process examples you can see here, they are represented as scripts. After these calculations, the user can add tasks for further data processing. This can also be done after measurement tasks, which you could also call data acquisition tasks. So state queries or measurement or data acquisition if you want. Here you can see a process model with a value that is measured and ähm subsequent regulation. So we have two scripts following the control event and then we have a service call which could be a Put or a Post request sending the new calculated manipulating value to a defined address. This is the data elements list I've mentioned. You can define global variables either as default values or as well as, which have to be overwritten during the process. And this list, you will see it a little bit later, but you can define the different endpoints you will be using for the service calls. For control you can also define interval frequency. Here also 0.1. You, as stated before, can define the control type, the value, the upper limit and the lower limit for the respective value. Wait means in this case again that for the next cycle to start, the system waits for the termination of all tasks. Also, for the regulation tasks. Sequential means that the tasks are executed one after the other. The state is first measured or queried and the regulation following this query is carried out with this measurement value. The current value V1 is subtracted from the optimum value V opt and the new control value MV is calculated with the difference. This is then sent with a service request to the corresponding actuator, which could be an element that actively influences the process. If you wish, you can combine the difference calculation, the control calculation, so the script of the PID, and of course also the sending of the command to the system into one sub-process. In the case of control, the type of control as well as the new control value and its limits can be entered, as I've shown you in the graphic. And if parallel were used here, the last value of V1 would be taken for the calculation, which would be no time guarantee in this case. State queries and regulations should be triggered at regular frequency. Termination events, however, cancel events, on the other hand, are triggered only by their termination or cancel conditions, which the user of course can define. An example for ähm cancel event or abort event would be when something triggers the abort of a cycle, such as for example, a watch dog function, to monitor the maximum cycle time or also like activating the emergency stop of the system. Here you can see a process with a value to be measured, a control and a termination condition. So we first have the measurement, we receive our value we want to process. Then we go to the control process flow. We calculate the difference between V opt and V1. We calculate the result for our PID controller and then we send the manipulated value to the actuator. And our emergency stop condition/ Sorry, our cancel condition would be if emergency stop is activated. So emergency stop dot active would be true. You can also add this as a global variable, emergency stop active. As a default, of course, it would be false. So as soon as the termination condition emergency stop active becomes true, repetitive tasks are terminated. The termination conditions are re-evaluated for each cycle. After the event is triggered/ And of course until they triggered the shutdown of the system. After the event is triggered, tasks for the cleanup routine can be processed before the cycle ends or the process is completely being terminated. You can see now a process where cleanup tasks have been defined. So after we have our measurement and our control cycle or control process, we can define after the cancel event is being triggered, also how we want to process later. Basically, you could include some simple shutdown routine for, for example, for your vessel one or for your system environment. The extensions presented are intended to help in the modeling of continuous processes by providing templates for the creation of process models, and of course, also on the other hand, through the representation as a closed loop subsystem with its own symbols for measurements, state queries, regulations and cancellation or termination events help to understand such processes a little bit more easily. In addition, for ähm clearer representation of the entire process, sub-processes can also be used for subdivision of the process flow of the model. Thus, we also want to fulfill the last characteristic we've defined before. Comprehensibility of the models for continuous processes. So now we come to examples for process models. I want to show you process examples modeled with the extensions presented in our paper. I would like you to take a look at the models and tell me what you can read out of them, what you can see. And whether the models fulfill the necessary information content for modeling the underlying control processes. So what you can read out of them and what you would identify, maybe as a lack of information or something like that. Beforehand, I will explain to you what kind of process is to be modeled for example, for the two of them. And I would like to ask you to give me open feedback on the models. #00:42:11-9#

34. B: Sure. #00:42:13-4#

35. I: So the first example is a simple feedback temperature control for a heat exchanger. The temperature of a liquid in a stirred tank is controlled using a heat exchanger. The heat flow introduced through the heat exchanger is controlled by a valve that controls the steam flow. The disturbing environmental influence to be considered is the fluctuating, the changing temperature of the liquid supplied. The tank is assumed to be insulated. And here is a another graphical representation of the process, in form of a process flow chart. So we have the inflow from above with the disturbing temperature fluctuation, we have the control of the valve here for the steam flow coming to the heat exchanger and we have the steering tank. The data elements listed here are used a little bit later for the script to calculate the PI controller. So you can define some static variables, some static information, data for the calculation, but you can also define variables which need to be overwritten in the process. And here we have a few examples for the endpoints. Of course these are not real, but for our example I've defined some, for example, for the temperature 1, for the vessel, for the tank. We have HTTP Get on localhost on a defined port, and then you follow just the value you have defined for the data server or you need to use to get the information you want. So our process model would look like this. We have defined wait and sequential. So the first edge shows again the measure event, which is being triggered once in a second. Then we want to get the Temperature 1 for temperature of the tank. Then we have, for example, if you want to include it, a conversion of the process value. If you don't need it, and if you can work with the value you get from the communication interface, of the sensor, it's fine and you don't need to model anything or include anything. On the other edge we have the measurement of the disturbance from the inflow. We assume that the inflow is constant, so we only need the temperature fluctuation. Then on the third edge we have the control event. So we apply on the values, we received from the system, we apply our PI model. Then if we want to include some kind of data conversion or if we need to perform some kind of conversion we can include it in form of a script as well. And then with the final result we can send the PI controller dot manipulating value directly to the actuator of the system which would be the motor of the valve in this case. And our stop condition. In the MathWorks library we don't have anything defined for ähm stop, for a cancel condition. So we introduce just some kind of basic process stop command. Stop activated. If it's set to true, we kind of execute some shutdown sequence in form of a script, but you can also include a service call or service call with script if you want. So this is basically just a representation of any shutdown routine you want to define here. Here we see an example of the data conversion. We just introduce some kind of renaming of a variable or copying of a value. The mathematical model is a little bit more complex, mathematical model of the PI controller. In this case we get our PI controller manipulating value and we can just transform the value, as I've told you before, in any way, you can define in a script. But here we just again copy the value and send this value to the following/ to the actuator

we've defined for this system, so yeah, to the endpoint we've included here. So this would be the basic process flow. We can again scroll ähm little bit up if you want to see it again, but the questions for the evaluation of this process model would be/ I would ask you to rate it according to the following criteria, on a scale starting from one, which would be very bad, to five, which would be very good, as stated here in the table. So the first thing would be comprehensibility. Following the process model you've just seen, would you say you, kind of, you know what happens during this process? #00:48:10-9#

36. B: Yes, sure. But previously, if I could ask you just the one question about the notation. Just to check if I understood it correctly. So basically, I got the part that we are waiting for a fresh task to be executed, and that's clear. For the sequential part this means that all those branches, let's say, those branches that are coming out the the gateway. Are they executed one after the other? #00:48:56-4#

37. I: Sequential in this case just means that the edge for the control event would be executed/ so the process on the edge modeled for the control event would only be executed following the tasks of the measurement events. In this case, in this process model, if these two are ready, if these two are completely being processed and are finished, this edge would be processed. #00:49:34-5#

38. B: Okay so it's like the control/ that process doesn't start until the measurement is finished. #00:49:45-2#

39. I: Exactly. #00:49:45-8#

40. B: Right? #00:49:46-6#

41. I: Yeah. #00:49:47-0#

42. B: Okay, I got it. I got it. Okay. #00:49:50-1#

43. I: So the calculation of the PI controller can only start if you get the //B: Yes, if/ // values from these two. #00:49:56-6#

44. B: If he has the right measurement. #00:49:59-6#

45. I: Exactly. #00:49:59-8#

46. B: Okay, I got it. #00:50:01-3#

47. I: The other case would be that you have some kind of maybe initial value or some kind of value from the last cycle of the closed loop system for T1 and four DT 2. Like for the temperature of the tank and this disturbance. And you would have to work with these two values, but you don't know exactly when they were collected from the system. So. #00:50:30-3#

48. B: Yeah I got it. It was just more on the graphical perspective. Let's say, since I'm more used to standard BPMN. So usually after the, let's say, everything that comes after the gateway or/ it is executed, just one of those branches, or all of them. So yeah, it's like kind of dependencies between branches that you're introducing, right? #00:51:07-8#

49. I: Only with this attribute, with //B: Yeah, yeah.// sequential. #00:51:12-3#

50. B: But only with that. But since it's uncommon in/ I mean, it's not something present in BPMN. So I was a bit/ I was trying to figure out if that was correct. Okay. I got it. Yes. (...) But based on that, I mean, clarified that part. I think everything is pretty clear and/ (...) #00:51:38-2#

51. I: So, regarding the rating. #00:51:42-0#

52. B: Yes, we can go back there. (...) Comprehensibility. I think it's pretty comprehensible. So I will rate it high. So. Four, five. I mean it's pretty comprehensible and a clear. Yes. #00:52:04-3#

53. I: Okay. #00:52:04-8#

54. B: It's pretty simple, but also, it's also/ this scenario is pretty simple, so I wouldn't expect less. (laughs) //I: Okay. // I wouldn't expect a difficult model from a single scenario, but/ #00:52:20-3#

55. I: So clarity would be, for you? #00:52:26-8#

56. B: I would rate it around four or five. I mean it's not a/ Yeah. #00:52:33-9#

57. I: And regarding the simplicity, would you say that you could represent the process, the model, a little bit more simple, even, being represented a bit a little bit more simply? #00:52:52-9#

58. B: You have compared to a possible standard version in BPMN or? #00:53:01-9#

59. I: Yeah, let's go/ Yeah. Let's take into consideration the basic standard. #00:53:09-9#

60. B: Yes, probably yes. Since you are introducing specific elements for your specific scenario. Yeah, probably yes. Usually/ I mean, when you want to model something that is specific in BPMN, you may end up with the/ even, let's say, a big model when it is really a simple scenario that you want to model. So probably it could be the same. So I think it's pretty simple, yes. #00:53:58-8#

61. I: So, you would consider using the extensions with other symbols you know from BPMN as being the more simpler way, the little bit simpler way to describe such processes? Or would you state that, regarding your knowledge about BPMN, you could think of a way to model these kind of processes even more simple without these extensions? Just to make it clear for me. #00:54:35-3#

62. B: Can you repeat it please? #00:54:38-9#

63. I: So just the short version would be, would you consider standard BPMN to be more effective or simpler to use to models such processes? Or would you consider BPMN together with the extensions we've defined here, or introduced here, to be the simpler way to model continuous processes? So which one is simpler for the user to apply, or the modeler to apply? #00:55:17-4#

64. B: Simpler for the user. It's difficult to say. I mean. The point is that, when someone uses BPMN, he usually uses the most common elements. So if you consider BPMN, also BPMN has already several elements that are really specific, but that nobody really uses. So I think it depends really on the case, on the scenario, that you are representing. If it is about continuous process, then probably it will be more/ for sure more effective using your presented extension. For, let's say, easy to use, it depends on the understanding that the user has of your extension, I would say. But for this specific case, and as I was mentioning before, sometimes even simple scenario that are not, let's say, standard business processes, they end up being difficult to represent. I had some experience of this, while we were trying to model some IoT scenario. And when you reach a level of, let's say, specificity that is not what is meant to be in BPMN. Then you find yourself struggling with several, with a lot of elements that lead to a really not understandable model at the end. So I think for this specific situation, if you target continuous processes problem, I think the support that you are going to provide, it should be valuable. Yeah. #00:57:37-9#

65. I: Okay, thank you. The next criteria would be logic. Is it clear for you what happens in parallel and what happens sequentially? #00:57:51-8#

66. B: I would say yes. Stated that, that I understood well the concept behind the gateway sequential and parallel stuff. #00:58:05-4#

67. I: So after a detailed explanation, you would say, it would be clear to you? #00:58:10-0#

68. B: Yes, yes. #00:58:11-3#

69. I: Okay, thank you. Then the final criteria would be extensibility. Could something be added to the model that would improve the information content? #00:58:25-2#

70. B: Difficult to say. I mean, for that specific scenario, I think that was/ I mean, there was everything that we needed to represent that scenario. So. I don't know. Honestly. I think that there was everything for that scenario, so I don't/ I'm not thinking about the possibility to add other information and I don't know what could be added. #00:59:04-3#

71. I: Okay, thank you. #00:59:07-1#

72. B: Probably someone that implements the actual scenario that actually works with the engineering part. Probably they could tell more on this side. #00:59:22-3#

73. I: Okay, thank you. I'd like to make a short break before we come to the next more complex model. So I'll stop the recording of the first part. #00:59:37-0#

INTERMISSION - 5 minutes break

74. I: Okay, recording started again. Okay, let's continue with the second of the two model examples. Here we have a model which is based on the description of the heating process taken from the training documents of the company Siemens. This is also a temperature control for a stirred reactor. The control is realized in this example with a PID controller, a manual control as well as a pulse generator. The heating is not done by a heat exchanger, but by a heating element. And furthermore the interlock conditions are defined in detail here. So we have kind of a basis to define them. The descriptions from the training documents for process modeling with Simatic PCS7 we used as a basis for this process model. It is modeled with an automatic control system that breaks out of the closed loop system when the system switches to manual control. This is our assumption. So we assume the system has already started and is controlled automatically and furthermore the process is described for only one reactor and not for two reactors. So yeah. Okay. (...) Here again we have some data elements being defined for interlock conditions. We have kind of a maximum temperature for the reactor. So we are not allowed to exceed sixty degrees. We have a minimum level for the reactor. 200 milliliters. And the other variables again are used for the calculation of the PID controller. And we also have some kind of default variables or default values for our interlock or cancel conditions. Like the operation mode is set initially to automatic instead of manual. The main switch is on. Stuff like that. The system works with cancel and sequential, which means that we have to stick to the defined duration of the cycles. And the model is a little bit more complex than the last model you've seen, so yeah. Here we have all together one, two, three, four, five, five state queries. Of course, the first measurement would be the temperature of the reactor. Then again, we have a service call combined with the script, if we want to model it like that to get the temperature of the reactor. Maybe also a data conversion. Next measurement would be the level. The actual level of the reactor. Because of course, this is also important for our system. Then we have the actual state of the operation mode. As I've stated before, it needs to be in automatic mode for this process. Then we have the actual state of the emergency stop, whether it's activated or not. And then we have the main switch. And after these five state queries we have the control edge again, PID controller in this case. We have a pulse generation and then we have again a service call which sends out the manipulated value, the process value, out to the respective actuator. The cancellation conditions would be the main switch if it is set to state off. In this case you could define, for example, a cleanup routine in form of sending a message to an operator. You could also add another task if you want to. The next cancellation condition would be if the emergency

stop would be activated, would be true. We have the temperature of the reactor, which is not allowed to exceed the maximum temperature as stated above. This would be here you go sixty degrees. And also of course we have a condition for the level. Minimum level would be 200 milliliters, so the actual level of the reactor is not allowed to go beneath this level, beneath the 200 milliliters. And the final condition would be, operation mode would change to manual, from automatic to manual and then we would need to call some kind of routine to change to manual operation mode and then the system would/ (...) Now I'm thinking about the term. Would/ The closed loop subsystem would stop and the complete process would either call another process model. Another process model would be triggered or the complete process would just be terminated. The mathematical representation of the PID controller is included in the script. It's kind of the same complexity as the PI controller. You just need to know the right, the correct mathematical representation, the mathematical model. So yeah, this model is a little bit more complex, but the evaluation will be the same as for the last model. So again, I will ask you about comprehensibility, about clarity, simplicity, logic and extensibility. And if it is okay for you, I can maybe scroll a bit, can zoom a little bit out to make it ähm little bit more visible for you. #00:06:37-0#

75. B: Yeah, but I got it. #00:06:49-4#

76. I: Okay. So if you want to go up, just say it and we can change the view again. So again, I ask you to rate the model according to the criteria here in this table, again starting with one, very bad, going up to five, very good. Starting with comprehensibility. Would you say that you perceive the sense, the basic process behind the model? Do you know what happens? #00:07:20-3#

77. B: Yes. And I understand what happens, in a/ Yes. I will rate high this criteria. #00:07:34-1#

78. I: Okay. #00:07:35-3#

79. B: So yeah. #00:07:36-6#

80. I: Clarity. Would you say, you can grasp the entire system at a glance? I know the system, this process, is a little bit larger, of course, because we have more conditions and more measurements. #00:07:50-9#

81. B: I mean, it's pretty clear, but it is obvious that an increasing amount of elements will affect somehow understandability, simplicity. But that's a matter, I mean, that's an already known fact. So it's not something related to let's say your extension. It is just what it is. The more amount of elements that are being used, the less understandable it gets, the model. So but, I mean, yeah, as I said. It is not related to the extension that you're proposing, but for sure, speaking about clarity and simplicity, I mean, it is pretty clear. But I would say, of course it is less clear and less simple than the previous one. But it is only a matter of amounts of element or notation elements that are being used. There is really a lot about understandability on BPMN models. Actually, we have also done something, some research about this. There are several metrics that can be used to define if model is/ I mean, you can see them as indicators. Indicators that tell you if the model will be received more or less understandable by the reader. But yeah. #00:09:39-2#

82. I: That's a good point. Thank you. Maybe I'll use it later in my work. #00:09:44-5#

83. B: Okay, I can show you some material if you want. #00:09:48-4#

84. I: That's great, thank you. It would really help, thanks. So you would say, of course increasing/ (...) yeah, increasing the number of elements, which need to be modeled or included in the process model, would also mean kind of a decrease in simplicity, of course. Would you say, it changes/ using the extensions, it changes at the same rate, so the complexity of the model would increase on the same rate as it would increase using the standard BPMN standard notation? #00:10:34-9#

85. B: I think it depends on the understanding that you have of the new notation or the extension. If you have the same understanding for the elements that you have for the common BPMN, then it doesn't make any difference. #00:10:52-9#

86. I: Okay. Yeah, we don't have an example purely modeled in BPMN right now, so yeah, of course. #00:11:03-8#

87. B: But then I/ it may be, I mean, it may be that what you want to express with your extension would require many other elements in BPMN. So it's difficult to say that/ I mean, probably, for sure there will not be a one-to-one match since you are adding, let's see, more elements that have more detail, I mean, different possibility, different thing to detail representing. So probably in those cases what you see, is that you will have, let's say, one element of your extension and several other of the BPMN. So even in that case you could also think about presenting your extension if/ In this perspective actually, you could say,'Okay, with BPMN to represent this scenario, I need, you know, thirty elements for/ With my extension. I just need ten.' Then it's only a matter of, if you understand those new elements that you introduce. And if you do, then your model is in some way simpler and more understandable. I would say. #00:12:38-9#

88. I: That's a good point. Thank you, yeah. Then the next characteristic or the next criteria would be logic. #00:12:51-8#

89. B: Yeah, logic, I mean, it's pretty much the same. I mean, it's not more difficult than before. I mean, if you have clear in mind the meaning of the sequential that is used so. Yeah, it's/ I don't think it is affected. #00:13:17-4#

90. I: Okay. #00:13:18-6#

91. B: And extensibility, as I said before, it's more/ I think more people from the engineering side that are actually working with those system could tell you more. From my perspective, the scenario is clear and there is everything that I would expect. #00:13:39-6#

92. I: Okay, thank you. Now to the more general questions. Based on these extensions, and let's also assume that you are as familiar with these extensions as you are with the standard BPMN. Would you be willing to implement this modeling method in your everyday work when you are working on modeling continuous processes? #00:14:09-8#

93. B: I would say yes. Especially if it allows me to reduce the amount of elements that I need to use for representing the same thing, the same concept. Yes, and it will result in me having a more clear (inc., insight?) on the same model. And also it would be easier to maintain probably. #00:14:38-3#

94. I: Okay, great. Thanks. Now there are a few questions, maybe a little bit difficult to answer, but if you want to/ Yeah, I'll leave it to you. #00:14:56-3#

95. B: Let's try. (laughs) #00:14:57-1#

96. I: How well do you think, these extensions describe a control system for these examples? So if you think about the basic, the real physical process, that is being modeled here, how well do you think would these models describe the process? For model one, the little bit simpler one, and for model two the second one, the more complex one. Again, with a rating from one to five. #00:15:30-2#

97. B: But, I mean, I guess, all of them reports all that is needed for representing those examples. So I will rate them high because I mean there is everything that I would expect. So there is the measurement, the control. And there is also the handling of a sub condition. So from my perspective, I think that there is everything that needs to be there. #00:16:05-2#

98. I: Okay, thank you. Then would you say, that there is something missing for a detailed process description or/? #00:16:14-7#

99. B: Nothing comes to my mind about missing details. Maybe/ No. It has everything automatized. So. No. #00:16:36-8#

100. I: Okay. Question number nine is really meant for control engineers. So yeah, I think you/ and I think you've already answered it, of course, as well. So we'll go right to number ten. Finally, we're coming to an end. I would like to go into a little bit more detail about the models with our extensions, and again ask you for a rating. Again, with the scale one to five, where five is the best value. We have a few questions here. So the first question would be, how easy is it to understand in the models shown that the individual processes run in parallel and independently of each other? #00:17:28-5#

101. B: Okay, but we have seen most, mainly example where, let's say, like the control was waiting for the previous one to happen, right? For the measurement to happen. We have seen mainly sequentials/ #00:17:51-1#

102. I: We have seen only sequential, yeah. #00:17:53-6#

103. B: Right, so, but so actually, if I could ask you a question about/ what if we had an example with the parallel, what would change? #00:18:07-8#

104. I: With parallel, you wouldn't have the (...) time constraints basically on the values you have measured in a cycle before. So with sequential, the basic idea behind it, is to use the most recent values, from the same cycle, to process them in the control edge, in the control process flow. So you have to use the last measured values from the sensors in your process model, or the state queries, and on the other hand parallel would mean that, you cannot guarantee with the time window, with the time frame, of the last cycle, when the last measurement has been performed and from which kind of time frame or from/ You cannot state exactly when the last measurement has been performed, and you need to use just the last available value for your state query or for your measurement, basically. So you have to work with the results from the last cycle in your control process. So this is the basic idea. It's basically meant for the cases, where you cannot easily define the time frame for your measurements in relation to your control answer. So if you cannot state in detail, how these two sequences relate to each other on a timeline, then you would use the parallel modeling version, the parallel attribute. And sequential is really to define how they are related to each other. #00:20:28-3#

105. B: Yeah I got it. I was just wondering, in which scenario I would use that, the parallel. But yeah, I got it. #00:20:40-6#

106. I: I mean, basically for the first process we don't have any time constraints. So for the easier process model, the simpler process model, you could have also used the parallel version. Because we don't have any information from MathWorks, when the values have to be measured or when the state queries have to be performed prior to applying the control algorithm. On the other hand, regarding the Siemens process model, we have the time constraints because it's an example for PLC (Programmable logic controller) cycle, for a program you use in PLCs, where you use on a daily basis, for example, the watchdog function, which kind of terminates your cycle if you're exceeding the defined cycle time. So the one system is not really constrained regarding duration, regarding cycle time. And the other system, of course, works basically using functions like watchdog functions like a defined cycle time. #00:22:00-1#

107. B: I got it. So coming to the question. (...) I would say that it is pretty easy then to understand. #00:22:18-4#

108. I: The second question would be, how easy is it to define, when an adjustment or a regulation is made or performed on a system? #00:22:28-7#

109. B: I mean, it's pretty easy. I mean, it's about the stop condition or the tasks that are conducted after the stop condition happened. So that's pretty easy. #00:22:45-6#

110. I: And regulation also means, of course, the answer regarding the control process flow. So based on the measurements, prior to the control or generally the measurements which collected data for the system. Based on this knowledge, the system would react with a defined controller algorithm, which on the other hand would be defined in these scripts you've seen here. #00:23:21-2#

111. B: Right. #00:23:22-1#

112. I: So yeah, regulation also means, what comes after the control event. #00:23:26-5#

113. B: Yes, after the control. Yes. (...) Yeah, then for for that part, I mean, it's pretty easy. It's straightforward to understand what is/ how those adjustments can be defined. #00:23:48-8#

114. I: Okay, thank you. The third question would be, how easy is it to define the maximum duration of an adjustment? Which means the cycle time basically. #00:24:05-8#

115. B: The cycle time, I'm guessing it could be a parameter that we set at the beginning, right? #00:24:12-7#

116. I: Yes, but respective to the control and the measurement event. I just want to show you again the graphic. For control events, it would be here, this value. So this is the, like, the definition window you would use to define exactly when your control event happens. And it's the same for the measurement event, basically. So in both cases we have the value interval frequency which is displayed in Hertz. #00:24:50-7#

117. B: So then I mean, it's pretty easy when you have the possibility to define it like we have just seen. It's not difficult. #00:25:13-5#

118. I: Okay, thank you. And the next question is, how easy is it to define, under which conditions all repetitive tasks shall end? So the cancel conditions, basically. #00:25:26-5#

119. B: But basically you define them in the model, right? #00:25:33-2#

120. I: Yeah, exactly, yeah. #00:25:34-4#

121. B: Yeah, it is what I was thinking about while answering the other question. Yeah, it's/ #00:25:47-4#

122. I: I just want to show you again the example where we stated it how to model/ Yeah, exactly so you would have a window just like this. You can state the condition in Ruby code. #00:26:00-6#

123. B: Yeah, but, while you are using the tool, for instance, you can interact with the model. So you click and you define, you click on the element and then you can define it there? #00:26:18-0#

124. I: Yeah, exactly. The [redacted] is not only an execution engine, it's also an editor. #00:26:22-6#

125. B: Yeah, so I mean, it's pretty straightforward if I can click on the element and define there the condition for the stop, it's pretty easy. #00:26:33-0#

126. I: Okay, great. Next question would be, how easy is it to define that cleanup tasks have to be done once after the system shuts down? So after the cancel condition is triggered. #00:26:50-2#

127. B: It's easy, you just place them after. (laughs) #00:26:55-9#

128. I: And the last question in this table would be, how easy is it to describe complex operations in the context of continuous processes with these extensions? Again, only meant to define the complexity of the modeling process itself for continuous processes. When you use the extensions. Is it complex or/? #00:27:26-3#

129. B: I mean, for defining complex operations, what do you mean exactly? #00:27:39-5#

130. I: For many people, it's not really intuitive how ähm control system, a closed loop system would work in production systems or in/ For example, like a simple example as a thermostat, as a temperature modulation for your home. It's like a simple example for you to comprehend, to know what's going on. But if you want to implement the control algorithm for it, you don't know exactly what values do you need to calculate the respective output or how, in which frequency would you measure the respective values or how can you define whether the system reacts/ (...) Are there any exceptions you need to handle or are there maybe any errors you need kind of to include in the model? Stuff like that. And all these relations, all this coming together in one process model could be really complex to kind of understand. And if you have only the specified point of view, only like the controller model itself, without any reaction of the system, without any exception handling or without any time constraints. Then you have you kind of know how the controller works, but you're missing the environmental information how everything works together. So it could be a little bit more complex than only the mathematical model of the controller. Because you have so many more relations, you need to consider in a model. So the question is basically, do the extensions help a process modeler or an engineering, a manager, help to understand the process behind applying a control algorithm? That's the basic question. #00:30:20-7#

131. B: Yes, I mean, I would say that it helps, (inc., #00:30:24-4#) to visualize all the, let's say, the measure, the values that are needed for conducting certain operations. So to decide what to do, and it gives you a good perspective of all the things that are involved in this scenario. So. I would rate it higher. #00:30:51-5#

132. I: So the target (goal) would basically be for us to give the modeler or the user an idea of what process values, what measurements are needed, what control loops are needed, and what kind of algorithms you need to apply to the system. And of course, what kind of interlock or cancel conditions you need to consider for your system to work. So this is the basic idea behind it. #00:31:17-2#

133. B: Right yes. I think it succeeds in representing all these aspects. So. #00:31:26-0#

134. I: Okay, thank you. Great. Yeah, thank you very much for taking the time to do this. We've finished our interview. I appreciate your feedback in any way on the guideline, on the duration of the interview, on the complexity of the questions, maybe also. Yeah, thank you very much. I'll start the recording now. Stop. Stop the recording now and if you want to, you can give me feedback if you like. #00:31:56-3#

135. B: Yes, why not.