

Transcript - Group 2 "Allrounders", Interview 2

I ... Interviewer (BLINDED)

B ... Expert

(Unv.)... Incomprehensible passage

(...) ... Pause longer than 3 sec.

() ... Comment

// ...// ... Speaker overlap

Transcript

1 1. I: Aufnahme läuft. #00:00:03-6#

2 2. B: Okay. #00:00:06-0#

3 3. I: Hallo und danke, dass Sie sich Zeit nehmen, um mit mir dieses Interview durchzuführen.

4 [REDACTED]
5 [REDACTED] Möchten Sie sich vielleicht
6 kurz vorstellen und Ihre Verbindung zu BPMN beziehungsweise zur Verfahrenstechnik,
7 Fertigungstechnik oder Prozessmodellierung erklären? Ich möchte Sie bitten, dabei nicht Ihren
8 Namen zu nennen, sondern nur die folgenden Informationen. Berufsbezeichnung und
9 Umschreibung Ihres Arbeitgebers, Basis Ihrer Expertise zum Forschungsthema, Ausbildung
10 beziehungsweise Ihr fachlicher Hintergrund und Ihre Berufserfahrung. Bitte. #00:00:51-4#

11 4. B: Ja. Hallo. [REDACTED]

12 [REDACTED]
13 [REDACTED]
14 [REDACTED]
15 [REDACTED]
16 [REDACTED]
17 [REDACTED]
18 [REDACTED]
19 [REDACTED]
20 [REDACTED]
21 [REDACTED]
22 [REDACTED] Ja, geht schon. #00:02:13-5#

23 5. I: Ja, das kann ich nachher auch herausstreichen. #00:02:15-8#

24 6. B: [REDACTED]

25 [REDACTED]
26 [REDACTED]
27 [REDACTED]
28 [REDACTED] #00:02:50-8#

29 7. I: Okay, super. Dankeschön. Kommen wir zur Einführung des Themas. Unsere Forschung
30 konzentriert sich auf die Entwicklung einer Methodik, um kontinuierliche Prozesse in BPMN
31 darzustellen und sie in einer Workflow Engine ausführbar zu machen. Für diese Aufgabe haben
32 wir an BPMN-Erweiterungen für kontinuierliche Prozesse gearbeitet. Warum kontinuierliche
33 Prozesse? Weil diskrete Prozesse bereits in anderen Forschungsarbeiten behandelt wurden und
34 nicht die gleichen Schwierigkeiten bei der korrekten Darstellung mittels BPMN aufweisen. BPMN
35 ist bereits ein weit verbreiteter Standard im Business Process Management und hat seinen Weg
36 in die Fertigung gefunden. Diskrete Fertigungsprozesse können bereits mit BPMN 2.0 modelliert
37 werden. Im Grunde wollen wir eine Methodik einführen, um solche Prozesse so darzustellen,
38 dass sie von jeder Person in einem Unternehmen, vom Ingenieur bis zum Manager, verstanden
39 werden können. Dies könnte durch die Verwendung dieser Notation erreicht werden. Ein
40 weiterer Vorteil ist auch, dass es bereits eine Reihe von Workflow Engines gibt. Das sind

Anwendungen, die die Ausführung dieser Prozessmodelle auf der Grundlage der für jedes Symbol implementierten Logik ermöglichen. Wir arbeiten mit einer webbasierten Anwendung, die erweiterbar ist und mehrere Kommunikationsschnittstellen implementiert hat. Ein weiterer Vorteil ist daher die Interoperabilität in diesem Zusammenhang im Vergleich zu anderen proprietären, starren Software-Anwendungen. Wir wollen herausfinden, ob diese Technik auch für die Implementierung von digitalen Abbildern eingesetzt werden kann. Da digitale Abbilder dazu dienen, ein physikalisches System oder einen Prozess in digitaler Form darzustellen, meist anhand von Daten oder mathematischen Modellen, mussten wir einen Weg finden, den Ablauf von kontinuierlichen Prozessen wie sie aus der Prozessindustrie bekannt sind, darzustellen. Aus diesem Grund haben wir uns auf die Modellierung von Regelkreisen konzentriert. Die Prozessmodelle sollen durch BPMN für Personen mit unterschiedlichem Hintergrund leicht verständlich sein. Die Interviews werden geführt, um herauszufinden, wie Prozess- und Regelungstechnik und Techniken aus der Business-Process-Modellierung kombiniert werden können und wie erste Ergebnisse von Experten wie Ihnen wahrgenommen werden. Außerdem wollen wir herausfinden, ob es Schwachstellen gibt, die von Experten identifiziert werden und wie wir diese beseitigen können. Noch ein paar Begriffe, die ich vorab erklären möchte. Der Begriff, digitaler Zwilling. Was verstehen wir darunter? Es gibt verschiedene Methoden, Dinge aus der echten Welt, zum Beispiel echte Maschinen, zu simulieren. Teils merkt man aber, dass es mehr Parameter brauchen würde als bei normalen Simulationsmethoden um eine Maschine vollkommen so abzubilden, wie sie sich in der Realität verhält. Bei einem digitalen Zwilling wird versucht, möglichst nahe an das reale Verhalten einer Maschine oder anderer Objekte heranzukommen. Das soll dazu führen, dass wenn etwas getriggert wird bei einer echten Maschine, der digitale Zwilling oder das digitale Abbild das gleiche oder ein möglichst ähnliches Verhalten zeigt. Was verstehen wir unter kontinuierlichen Prozessen? Kontinuierliche Prozesse möchte ich mit Beispielen vielleicht erklären. Wenn man Bierbrauen her nimmt, gibt es zwei Möglichkeiten. Die diskrete, also nicht-kontinuierliche Variante wäre, wenn man in einem geschlossenen Kessel die Zutaten hinein gibt, zehn Liter Wasser und so weiter, und den Brauprozess einfach schrittweise ablaufen lässt. Am Ende kommt dann eine begrenzte Menge an Bier heraus. Die andere kontinuierliche Variante wäre, wenn man keinen vollkommen abgeschlossenen Kessel hat, sondern miteinander verbundene Kessel, bei denen immer wieder Zutaten zugefügt werden und immer wieder Bier entnommen wird. Das geht die ganze Zeit so, sodass man nicht nachvollziehen kann, welcher Liter Wasser zu welchem Liter Bier gehört. Dabei läuft ein Teilprozess im ersten Kessel ab während gleichzeitig im letzten Kessel der letzte Prozessschritt stattfindet, bevor das Bier fertig wird. Und der letzte Begriff, geschlossener Regelkreis. Ein geschlossener Regelkreis, ist jene Logik in Form von Hardware oder Software, die das kontinuierliche Bierbrauen ermöglicht. Wenn man einen Prozess wie das kontinuierliche Bierbrauen hat, muss man schauen, wie man schlechtes Bier vermeidet, während der Prozess läuft. Man möchte die Qualität auf einem gewissen Punkt halten. Beim schrittweisen Bierbrauen hat man nur die zehn Liter, bei denen etwas schiefgehen kann und mit den nächsten zehn Litern macht man es dann besser. Aber was ist, wenn man die Brauanlage dauernd laufen lässt und ständig Bier austritt? Dann muss man währenddessen den Prozess überprüfen und schauen, dass man die gute Qualität des Bieres erhält. Das heißt, man testet oder misst Werte, die die Qualität beschreiben, überprüft, wie sich diese Werte von optimalen Werten unterscheiden und reagiert dann entsprechend. Stimmt etwas beim Zucker- oder Alkoholgehalt beispielsweise nicht, muss das Mischverhältnis geändert werden. Das heißt, in einem geschlossenen Regelkreis werden, während der Prozess läuft, gewisse Werte überprüft. Diese werden mit optimalen Werten verglichen und je nach Abweichung reagiert das System darauf. Kommen wir nun zu den Fragen. Aus Informatik-sicht bestehen kontinuierliche Prozesse aus einer sich ständigen wiederholenden Abfolge von Zustandsabfragen und Regulierungen. Zustandsabfragen und Regulierungen sind jeweils traditionelle Code-Stücke, die sich auf Sensoren oder Aktuatoren beziehen. Um solche kontinuierlichen Prozesse konsistent formal zu beschreiben, zu modellieren und in weiterer Folge ausführen zu können, haben wir eine Liste von Merkmalen identifiziert. Frage Eins. Würden Sie die Merkmale für wichtig oder unwichtig einstufen? Und ich würde Sie bitten, dass Sie Ihre Antwort auch begründen. Die Merkmale, die wir hier aufgelistet haben, sind Nummer Eins. Verschiedene Zustandsabfragen- und Regulierungskombinationen sind unabhängig und können parallel ablaufen. Nummer Zwei. Regulierungen folgen immer auf Zustandsabfragen. Nummer Drei. Die Dauer von jeder Zustandsabfragen- und

98 Regulierungskombination ist beschränkt. Nummer Vier. Wenn Zustandsabfragen gewisse
 99 Ergebnisse liefern, wird das System beendet. Nummer Fünf. Bevor das System beendet wird,
 100 muss es in einen konsistenten Zustand gebracht werden. Und Nummer Sechs. Das resultierende
 101 System soll für Menschen verständlich sein. Gut. Nummer Eins. Verschiedene Zustandsabfragen-
 102 und Regulierungskombinationen sind unabhängig und können parallel ablaufen. Würden Sie
 103 sagen, das ist wichtig oder unwichtig? Und bitte//Also.// begründen Sie Ihre Antwort.
 104 #00:10:01-1#

105 8. B: Ja, also, ich würde ganz klar sagen, dass es meiner Meinung nach wichtig ist. Der Grund ist,
 106 wenn ich schon so einen kontinuierlichen Prozess habe, ja, dann habe ich wahrscheinlich
 107 mehrere Messungen. Weil wenn ich nicht mehrere Messungen habe, dann habe ich ja schon fast
 108 einen diskreten Prozess, ja. Das heißt, ich habe vielleicht, wenn man jetzt das Bierbrauen her
 109 nimmt, den Flow zwischen diese einzelnen Kessel und ich werde da wahrscheinlich überall
 110 verschiedene Messungen haben. Und wenn die aber nicht parallel ablaufen können, wie soll ich
 111 dann Entscheidungen treffen können, ja. Also, sagen wir, es fließt vom ersten Kessel in den
 112 zweiten und da möchte ich vielleicht schon Anpassungen vornehmen, damit ich eben dann im
 113 zweiten Kessel schon Anpassungen habe an den ersten Kessel. Oder den ersten Kessel anpassen,
 114 dass dann eben kein schlechtes Bier herauskommt. Also ich glaube, Parallelität und
 115 Unabhängigkeit ist wahnsinnig wichtig, damit man eben die Regulierung überhaupt richtig
 116 durchführen kann für diese einzelnen Schritte. Weil wenn ich das nicht parallel machen kann, auf
 117 was für einer Basis soll ich dann die Regulierung durchführen? #00:11:24-3#

118 9. I: Danke. Zweite Eigenschaft. Regulierungen folgen immer auf Zustandsabfragen. #00:11:33-1#

119 10. B: Vielleicht gebe ich mal meine Erklär/ Also vielleicht frage ich mal vorher. Eine Zustandsabfrage.
 120 Ja, da oben waren das Sensoren und so weiter. Meine Frage ist, weil, wenn ich zum Beispiel als
 121 Externer jetzt in diesen Kreislauf eingreifen möchte, weil ich sage 'Ja, es kommt das Oktoberfest,
 122 wir machen jetzt doch schnell ein anderes Bier.' Gilt das dann als Zustandsabfrage? Weil ich
 123 möchte aktiv eine neue Regulierung einführen. Ja? Und ich glaube, dass es notwendig sein
 124 könnte, aber da bin ich mir eben nicht sicher, dass ich die Regulierung auch auf andere Seite, auf
 125 andere Weise triggern kann, wie nur durch eine Zustandsabfrage. Und da weiß ich halt nicht ob,
 126 wenn ich jetzt von außen eben das ändern möchte, ob das als Zustandsabfrage gilt. Wenn ich
 127 jetzt sage 'Das Bier ist zu bitter für ein Oktoberfest.' Ist das dann eine Zustandsabfrage? Oder
 128 würde das dann nicht unter Zustandsabfrage fallen? #00:12:44-8#

129 11. I: Sie meinen, wenn von außen ein Eingriff erfolgt? #00:12:51-9#

130 12. B: Genau, genau. Wenn ich als Mitarbeiter trotzdem sage, 'Das passt so nicht.' Wenn ich sage,
 131 'Ich weiß es besser wie diese Zustandsabfrage.' Weil eben sich noch andere Gegebenheiten
 132 geändert haben. #00:13:06-7#

133 13. I: Also, der Charakter hinter einem geschlossenen Regelkreis ist, dass man den Set Point, also
 134 den Zielwert, definiert. Und sollte sich/ also das System würde dann so funktionieren, dass es
 135 versucht, aufgrund des Reglermodells, das es intern schon kennt, das es von sich aus kennt, also
 136 die mathematische Operation, die dann auf einen gewissen Wert, den man abfragt, angewendet
 137 wird, eine entsprechende Antwort und Reaktion liefert. Und eine/ also zum Beispiel, wenn das
 138 Bier zu bitter ist und von außen dann ein Eingriff erfolgt, wäre es eigentlich eine Anpassung des
 139 Set Points, des Zielwertes. #00:13:51-8#

140 14. B: Des Zielwertes. Dass ich sage, 'Okay, der neue Zielwert ist jetzt Oktoberfestbier.' Und
 141 entsprechend würden sich auch die Regulierungen dann automatisch ändern, weil wir haben
 142 jetzt andere Parameter, die wir erreichen müssen. #00:14:05-4#

143 15. I: Genau, also im Grunde vom Charakter her würde ich jetzt persönlich sagen, dass sich dann
 144 generell der Prozess ändert und dass es sich dann um ein anderes Prozessmodell auch handelt.
 145 Weil einfach die Anfangsbedingungen, also der Set Point, der Zielwert, ein anderer wird.
 146 #00:14:22-7#

- 147 16. B: Ja, dann. Also, dann sehe ich eigentlich keinen Grund, warum die Regulierung nicht auf eine
148 Zustandsabfrage folgen sollte. Moment. (...) Folgen heißt aber nicht, dass sie sofort darauf folgen
149 müssen, oder? Weil das könnte auch/ Wir haben ja vorhin gesagt, es gibt Unabhängigkeiten und
150 Parallelitäten, ja? //I: Ja.// Und natürlich können ja in dem Kreislauf von oben, wenn man jetzt
151 nochmal auf das Bierbeispiel her geht, ja? Könnten ja verschiedene Zustandsabfragen zu
152 verschiedenen Regulierungen, vielleicht am gleichen Ort, führen und vielleicht muss man auch
153 die Zustandsabfragen, wenn sie jetzt alle parallel ablaufen, einmal abwarten, ja? Und DANN
154 entsprechend erst die Regulierung durchführen. Also, ich würde schon nach der Zustandsabfrage
155 eine Regulierung machen, //I: Ja.// aber vielleicht muss man auch festlegen, dass eben da
156 mehrere Zustandsabfragen zu einer Regulierung oder eine Zustandsabfrage zu mehreren
157 Regulierungen führen könnte. Vielleicht. Wenn ich jetzt wirklich am Schluss feststelle, 'Uh. Das
158 Bier ist zu bitter. Da muss ich vielleicht im Kessel Eins und Drei eine Regulierung durchführen,
159 vielleicht.' Also ich denke schon, dass das eine zum anderen führt. Also ich sehe das schon als
160 wichtig an und als eigentlich gesetzt, dass eine Zustandsabfrage/ ich meine, wenn das Ergebnis
161 passt, dann muss natürlich keine Regulierung stattfinden, ja? Das ist aber eh klar. Aber
162 ansonsten wenn es nicht meiner Erwartung entspricht, dann denke ich schon, dass eine
163 Zustandsabfrage zu einer Regulierung führt. Und in dem Fall ist es dann auch wichtig, weil wenn
164 ich dann die Regulierung nicht durchführe, dann werde ich das Produkt wahrscheinlich kaputt
165 machen und dann ist es unbrauchbar. #00:16:53-0#
- 166 17. I: Ja, das stimmt, ja. Es ist ein guter Punkt, den Sie hier angesprochen haben, aber wir kommen
167 ein bisschen später im Interview auch auf diese Bedingungen zu sprechen. #00:17:05-3#
- 168 18. B: Okay, ja. #00:17:06-7#
- 169 19. I: Und die Frage ist/ Die Frage hätten wir uns so gedacht, dass sie darauf abzielt, zu hinterfragen,
170 eigentlich mehr auf das, was Sie am Anfang eingegangen sind. Ob man generell den aktuellen
171 Zustand des Systems kennen sollte, bevor man eine Anpassung anbringen kann und ob das
172 wichtig ist? #00:17:30-7#
- 173 20. B: Ja, natürlich muss ich den aktuellen Zustand kennen. //I: Das war eigentlich/ // wenn ich den
174 nicht kenne, dann/ Ja. #00:17:41-9#
- 175 21. I: Selbstverständlich. #00:17:45-2#
- 176 22. B: Ja. #00:17:46-5#
- 177 23. I: Darauf war es eigentlich abgezielt. Okay, ja, Dankeschön. Punkt Drei. Die Dauer von jeder
178 Zustandsabfragen- und Regulierungskombination ist beschränkt. Beziehungsweise beschränkt
179 könnte man auch ersetzen durch das Wort definiert. #00:18:12-1#
- 180 24. B: Okay. (unv.) (...) Das ist ein bisschen schwierig das Ganze. (...) Also vor allem im weiteren
181 Sinne, wenn ich jetzt sage, 'Das ist beschränkt', ja, dann muss ich mir ja auch Gedanken darüber
182 machen was ist, wenn ich diese Schranke überschreite. #00:19:08-7#
- 183 25. I: Ja. #00:19:10-2#
- 184 26. B: Und das glaube ich macht es ein bisschen komplex, das Ganze. Wenn ich jetzt sage, 'Die Dauer
185 ist immer unbeschränkt', ja, also quasi das Gegenteil von dem da, dann hab ich wiederum das
186 Problem, was mache ich, wenn die Abfrage und Regulierung einfach nie aufhört? Weil gerade
187 bei der Abfrage ein Fehler unterlaufen ist und es deswegen nie zur Regulierung kommt. Also. Die
188 logische Antwort ist natürlich, dass das beschränkt sein muss in irgendeiner Art und Weise. Weil
189 es sonst ja nie meinen Prozess voranschreiten kann. Wenn ich das nicht irgendwie beschränke.
190 #00:19:58-4#
- 191 27. I: Ja. #00:20:00-9#
-

- 192 28. B: Ist das wichtig oder/ Also. Im Kontext der andere Eigenschaften (...) ist mir das jetzt/
193 Grundsätzlich ist das wahnsinnig wichtig, weil wenn also/ Aber die Frage ist, ist es wichtiger oder
194 unwichtiger. Ich sehe auch das als wichtig an. Also. Es mag blöd klingen. Irgendwie sind alle
195 Sachen wichtig. Aber nur wenn ich nicht weiß, ob das irgendwie beschränkt ist oder definiert ist,
196 wie lange das dauern darf, dann komme ich nie weg von dem Ding, ja, also, von einer
197 Zustandsabfrage oder Regulierung. Das sehe ich sonst ein bisschen als Problem, wenn das nicht
198 beschränkt ist. //I: Okay, also/ // Außer man sagt, 'Es ist beschränkt', ja, und man darf aber
199 trotzdem, wie soll ich sagen, Workarounds machen, die dann sagen, keine Ahnung, nach zehn
200 Sekunden, 'Mach noch einmal die Zustandsabfrage, weil da kam nichts retour', ja. Das ist die
201 Frage, ob das dann auch zählt. (...) Aber an und für sich muss es einen fixen Zeitpunkt geben,
202 dass dieses Ding auch aufhört und man auch eine Zustandsabfrage gekriegt hat und eine
203 Regulierung durchgeführt hat. #00:21:45-8#
- 204 29. I: Okay. Danke, ja. Das ist eine sehr gute Begründung gewesen. Dankeschön. Nummer Vier.
205 Wenn Zustandsabfragen gewisse Ergebnisse liefern, wird das System beendet. #00:22:06-4#
- 206 30. B: Wenn Zustandsabfragen gewisse Ergebnisse liefern, wird das System beendet/ Ja,
207 wahrscheinlich. Also, wenn die Zustandsabfrage jetzt sagt, der Kessel ist leer, dann werde ich
208 sagen, 'Okay. Für heute ist Ende, Feierabend.' Nein, also. Oder wenn die Zustandsabfrage mir
209 sagt, 'Okay, das Bier ist einfach nicht mehr zum Retten. Wir haben einfach schon so viel falsch
210 gemacht', dann muss ich wahrscheinlich das System beenden und den Kessel putzen. Also es
211 wird notwendig sein, wenn man den Zustand abfragt und merkt, okay, es gibt keine, zum Beispiel,
212 Regulierung mehr, die das Bier rettet. Ja, dann wird man das System wahrscheinlich beenden.
213 Oder wenn/ ich weiß nicht, ob es da noch andere Zustandsabfragen gibt, die dorthin führen
214 können, wie eben, der Kasten ist leer, oder, es sind acht Stunden vorbei. Wir dürfen einfach
215 nicht länger Bier brauen wie acht Stunden am Tag, ja. Deswegen wird das System beendet, ja,
216 ich glaube so eine Zustandsabfrage kann im Prinzip alles sein. Also, ja, es ist schon wichtig, dass
217 eine Zustandsabfrage irgendwie zum Shutdown führt, von dem Ganzen. #00:23:31-1#
- 218 31. I: Okay. Nummer Fünf. Bevor das System beendet wird, muss es in einen konsistenten Zustand
219 gebracht werden. #00:23:42-6#
- 220 32. B: Also ich glaube, wenn man schon mal etwas irgendwie mit Maschinen gearbeitet hat, dann
221 erachtet man das immer als eines der wichtigsten Güter. Der Grund ist natürlich, wenn man
222 beim nächsten Mal wieder den Bierbrauprozess startet und vorhin nicht weiß, wie/ und vorher,
223 weiß ich nicht, einmal hat man fünf Liter Bier drinnen gelassen, einmal hat man zehn Liter Bier
224 drinnen gelassen, dann hat man wieder gar nichts drinnen gelassen. Wie auch immer, ja. Bier
225 wird natürlich schlecht, wenn man es einfach so drinnen lässt, ja, aber an und für sich, man wird
226 den Kessel immer leeren oder entweder immer voll lassen, aber halt immer gleich, immer
227 konsistent. Weil sonst kann man beim nächsten Mal einfach nicht weitermachen. Ohne, dass
228 man wieder Messungen durchführt, aber das möchte man ja eigentlich vermeiden. Also, vor
229 allem, weil man es ja leicht vermeiden kann, indem man immer den gleichen Zustand erzielt.
230 Man macht trotzdem/ Beim nächsten Start wird man trotzdem diverse Messungen durchführen,
231 aber an und für sich sollte der Zustand schon immer konsistent sein. #00:24:54-6#
- 232 33. I: Der letzte Punkt. Das resultierende System soll für Menschen verständlich sein. System können
233 Sie hier ersetzen durch den Begriff Modell, auch gerne. #00:25:14-3#
- 234 34. B: Okay, weil ich hätte gesagt, das System DAHINTER muss nicht unbedingt verständlich sein,
235 zumindest nicht für jeden Menschen. Aber das Modell sollte natürlich für die Leute, die da
236 involviert sind, sollte das verständlich sein. Und der Grund ist, wenn ich das Modell nicht
237 verstehe, wie kann ich dann das Modell überhaupt entwickeln? Ja. Weil dann entwickle ich ja
238 etwas, das was ich selbst nicht verstehe, und wie soll das ablaufen? Hoffe ich einfach, dass es
239 das tut? Hoffe ich einfach, dass die Regulierung genau das macht, was ich als Ergebnis von der
240 Zustandsabfrage zurück bekommen habe? Weil dann muss ich hoffen, wenn ich es nicht
241 verstehe, dann kann ich einfach nur darauf hoffen, dass das funktioniert. Und deswegen denke
242 ich schon, dass Menschen, zumindest einmal die, die es natürlich entwickelt haben, müssen es

243 verstehen, und vielleicht auch die, die das verwenden, aber die müssen/ Die sind eigentlich eh
 244 so in den Prozess eingebunden, dass die das Modell vielleicht nicht verstehen können müssen,
 245 aber zumindest verstehen müssen, was sie tun müssen. Also zum Beispiel der Bierbraumeister,
 246 der muss jetzt wahrscheinlich den ganzen Prozess/ er wird ihn inne haben, aber muss vielleicht
 247 das Modell so gar nicht verstehen. Sondern wenn eine Zustandsabfrage kommt, dann kann er
 248 die verarbeiten und verstehen und er kann auch die Regulierung verstehen, aber er muss nicht
 249 genau/ den Rest dahinter muss er nicht verstehen. Aber die Leute, die den Prozess entwickeln,
 250 die müssen da von A bis Z eigentlich das Ganze verstehen können. #00:27:07-2#

251 35. I: Okay, Dankeschön. Von dieser letzten Eigenschaft kommen wir zur zweiten Frage. Können Sie
 252 grafische Eigenschaften nennen, also für das Modell, die Sie für die Modellierung
 253 kontinuierlicher Prozesse wichtig finden? Und ergeben sich daraus vielleicht Merkmale, die wir
 254 hier in dieser Liste, also außer diese sechs Merkmale, vielleicht vergessen haben? #00:27:38-6#

255 36. B: Okay, also die Parallelität haben wir schon gehabt. Ich denke, also, wenn ich jetzt speziell an
 256 BPMN denke, ja, dann denke ich auch, dass eine Loop sicher notwendig ist, ja, weil ich habe ja
 257 kontinuierliche Zustandsabfragen und Regulierungen. #00:28:00-0#

258 37. I: Ja. #00:28:00-6#

259 38. B: Das heißt, die werde ich da auch definitiv brauchen und etwas/ Das wird jetzt vielleicht ein
 260 bisschen schwer zum erklären. Wenn ich jetzt mehrere Messungen parallel habe, ja, dann muss
 261 ich auch irgendwie verständlich machen, wohin/ also, was für einen Effekt diese Messungen
 262 haben werden. Also die Zustandsabfragen, ja, zu welcher Regulierung die führen können, ja, weil
 263 ich habe zum Beispiel, wenn ich jetzt fünf Kessel habe beim Bierbrauen und ich hätte nach jeden
 264 Kessel, wo das durch geschleust wird, ja, habe ich eine Zustandsabfrage, ja. Und das kann, also,
 265 heißt, ich habe fünf Messungen, aber vielleicht habe ich nur drei Regulierungsstationen. Oder
 266 vielleicht habe ich eh fünf, ist ja wurscht, ja. Aber eventuell muss auch ersichtlich sein, welche
 267 Zustandsabfrage zu welcher Regulierung führt. Und das könnte wichtig sein, ja. Wenn ich jetzt
 268 sage, im letzten Schritt beim fünften Kessel merke jetzt, 'Oh, da müssen wir ein bisschen nach
 269 regulieren.' Aber das kann ich nicht nur im fünften Kessel machen, sondern muss vielleicht auch
 270 im zweiten Kasten ein bisschen etwas regulieren, ja. Vielleicht ist, ich meine, ich weiß es nicht, ja.
 271 Ich habe da auch vom Bierbrauen zu wenig Ahnung. Also ich schätze, wenn ich im fünften Kessel
 272 eine Abfrage mache, ist die Regulierung nur für den fünften Kessel relevant, aber es könnte ja
 273 Anwendungsgebiete geben, wo ich auch den zweiten Kessel regulieren muss. Und da ist die
 274 Frage, ob man so etwas abbilden kann. Und dann etwas anderes, was vielleicht aus
 275 Modellierungs- und Verständnissicht sinnvoll sein könnte, ist, wenn ich die Zustandsabfragen
 276 vielleicht so anordnen könnte, wie sie passieren, ja. Dass ich jetzt habe, es ist die Abfrage für den
 277 ersten Kessel, dann die Abfrage für den zweiten Kessel, also. Und nicht irgendwie, wenn ich mir
 278 jetzt wirklich ein BPMN-Modell, eine Parallelität vorstelle, dann ist das nicht geordnet, ja. Das
 279 heißt, ich kann die Branches verschieben, wie ich möchte, ja. Und die Frage ist, ob man das
 280 vielleicht ein bisschen geordneter machen kann, wenn man weiß, dass man ja zwei
 281 Zustandsabfragen hat, die eigentlich parallel stattfinden, aber der Prozess selbst ist ja schon
 282 irgendwie eine Sequenz. Ist das verständlich? #00:30:48-0#

283 39. I: Ja. #00:30:49-5#

284 40. B: Ja. Das ist eben die Frage, ob man das kann. Ich wüsste jetzt nicht wie, außer bei den Branches
 285 das dazu schreiben, aber dass das jetzt vielleicht Schritt Eins ist, das ist Schritt Zwei. Oder
 286 definieren, dass die Branches, keine Ahnung, von unten nach oben dann quasi sequentiell sind,
 287 oder von oben nach unten, oder/ Nur ich glaube, dass das für das Verständnis praktischer ist,
 288 weil diese Anordnung der Branches //I: Ja.// dem entspricht, was ich mir von der Realität
 289 erwarte und nicht, dass mal kommt, die Abfrage für den fünften Kessel, die Abfrage für den
 290 dritten, die Abfrage für den vierten, die Abfrage für den ersten Kessel. Weil dann werde ich
 291 durcheinander kommen. #00:31:38-5#

292 41. I: Ja, ich verstehe, ja. Dass man auf den ersten Blick auch sehen kann, in welcher Reihenfolge das
293 abläuft. #00:31:48-7#

294 42. B: Genau, ja. Grundsätzlich, ja, natürlich. Man hat ja durch den kontinuierlichen Prozess so nicht
295 unbedingt die Reihenfolge, aber de facto wenn ich in der Fabrik stehe, gibt es schon eine
296 Reihenfolge, wie das Bier durchfließt. #00:32:05-0#

297 43. I: Ja. #00:32:05-7#

298 44. B: Und das könnte halt auch ein komplexes Thema sein. Würde aber sicher da, wenn man
299 dadurch eine zusätzliche Komplexität einführt, bei der Darstellung, würde das aber sicher helfen,
300 dass man das Modell leichter verstehen kann. Da muss man/ Das ist glaube ich ein schmaler
301 Grad zwischen, man führt zusätzliche Komplexität ein, und, man erleichtert es dem Anwender.
302 #00:32:36-3#

303 45. I: Damit hätten Sie eigentlich auch schon die nächste Frage ein bisschen //B: Ja, ja. //
304 beantwortet. #00:32:45-3#

305 46. B: Ja, ich habe es mir gerade gedacht. Das ist eigentlich auch schon eben guter Teil von dieser
306 nächsten Frage, genau. #00:32:52-1#

307 47. I: Ja, also, wo liegen Ihrer Meinung nach die Herausforderungen, solche Prozesse abzubilden? Ja.
308 #00:32:58-9#

309 48. B: Ja, eben. Das ist nämlich auch das mit eben, welche Zustandsabfrage führt zu welcher
310 Regulierung. #00:33:07-4#

311 49. I: Ja. #00:33:07-5#

312 50. B: Ja. #00:33:08-4#

313 51. I: Okay, dann können wir eigentlich eh gleich weitermachen. #00:33:12-7#

314 52. B: Ja. #00:33:14-4#

315 53. I: Und zwar. Wir kommen jetzt zu den Extensions. Ich werde Ihnen Prozesse zeigen, die mit
316 BPMN 2.0 und mit unseren Extensions, Erweiterungen, modelliert wurden. Die Erweiterungen
317 sollen zum einen vordefinierte Modellierungskonventionen für in der Prozess- und
318 Steuerungstechnik übliche Routinen bereitstellen und zum anderen helfen, die Unterschiede
319 zwischen den parallelen Pfaden in den Prozessmodellen zu visualisieren. Die Prozesse werden in
320 der [REDACTED], modelliert. Und soweit ich weiß, sind Sie schon
321 mit der [REDACTED] //B: Ja, kenne ich. // Sehr gut, sind Sie schon vertraut. Das heißt, die Erweiterungen,
322 die speziell für die [REDACTED] schon eingeführt wurden, also Script, Service Calls, und so weiter, sind
323 Ihnen auch bekannt? #00:34:08-6#

324 54. B: Das Script und so/ Ja, die gibt es schon/ Ja, das gibt es schon lange. #00:34:13-8#

325 55. I: Okay, wunderbar. Dann kommen wir zu unseren Extensions im Zuge dieser Arbeit. Die erste
326 Erweiterung ist ein Gateway, das Closed Loop Subsystem Gateway. Dieses Gateway ist eine
327 Kombination aus einem inklusiven und einem ereignisbasierten Gateway. Es enthält
328 Verzweigungen beziehungsweise Kanten, die für die Zustandsabfragen- und Regulierungsphasen
329 des Zyklus ausgelöst werden, sowie Verzweigungen, die beim Empfang von Abbruchereignissen
330 ausgeführt werden. Die Ereignisse und Tasks in den einzelnen Kanten sind unabhängig
331 voneinander. Damit erfüllen wir das erste der oben genannten Features, dass einzelne Verläufe
332 unabhängig voneinander sind und sie parallel ausgeführt werden. Das Gateway ermöglicht
333 außerdem die Definition der Intervalldauer jedes Zyklus, sowie von Überschreitungsbedingungen,
334 beispielsweise hier wait oder cancel. Und der Ausführungsreihenfolge für Zustandsabfragen und

335 Regulierungen beziehungsweise könnte man das auch als Mess- und Steuerungsaufgaben
336 bezeichnen. Diese werden durch parallel und sequential gekennzeichnet. Das heißt, wir haben
337 hier die Attribute Interval duration overrun als cancel oder wait. Und Measure control cycle
338 execution, parallel oder sequentiell. Ich würde jetzt gerne diese zwei Attribute erklären, und
339 zwar einerseits wait oder cancel. Wenn wait gewählt wird, beginnt die nächste Iteration, wenn
340 alle Verzweigungen beendet sind und die festgelegte Intervalldauer erreicht ist. Bei cancel
341 definiert die Intervalldauer genau die Zeit, in der jeder Zweig zu beenden ist. Wenn die Tasks in
342 einem Zweig schneller beendet werden, wird der Zweig warten. Wenn noch nicht alle Tasks
343 beendet sind, werden sie abgebrochen. Das heißt, wait würde warten, cancel würde abbrechen.
344 #00:36:18-8#

345 56. B: Okay, das hört sich nach einem Problem an, das Cancel. (lacht) #00:36:24-9#

346 57. I: Okay. #00:36:29-0#

347 58. B: Also, vielleicht sage ich das gleich. //I: Ja. // Also das Cancel, das bricht dann ab und man
348 kommt quasi wieder, man fängt wieder von vorne an, macht wieder ein Measure und dann wird
349 das Ganze wieder ausgeführt oder? #00:36:43-5#

350 59. I: Genau, also. // B: Grundsätzlich. // Im Grunde geht es darum, dass wenn eine gewisse Dauer
351 für einen Strang vorgegeben ist, also, das wird durch weitere Erweiterungen, die wir hier
352 definieren, auch genauer beschrieben, durch Measure und Control Events. Wenn diese
353 Zeitbedingung überschritten wird bei cancel, dann würde einfach der neue Zyklus wieder los
354 starten. Bei wait würde man warten, ob diese Tasks, die dann nach den Events modelliert
355 wurden, auch wirklich beendet werden. Also man würde darauf warten, dass die beendet
356 werden. #00:37:23-2#

357 60. B: Also, wenn wir jetzt beim Bierbrauen nämlich sind, ja, und ich dieses Measure durchführe.
358 Und dann macht die Regulierung, da wird mehr Zucker hineingekippt, ja. Und dann kommt aber
359 ein Cancel, weil das schon zu lange braucht. Dann breche ich den ab, mitten beim
360 Zuckerhineinkippen, ja. Mache das quasi wieder zu und es kommt kein Zucker mehr dazu. Und
361 dann kommt die nächste Zustandsabfrage und wieder Regulierung. Ich glaube, das könnte
362 vielleicht problematisch werden. Ich weiß es nicht, aber da würde man vielleicht dann eben wait
363 verwenden und nicht cancel, also. #00:38:08-9#

364 61. I: Man müsste es auf den jeweiligen Prozess anpassen, ja. #00:38:12-0#

365 62. B: Okay. #00:38:16-8#

366 63. I: Ja. Kurz noch zu parallel und sequentiell. Bei parallel werden die Tasks nach Measure und
367 Control Events parallel ausgeführt. Bei sequential werden die Tasks nach Control Events erst
368 ausgeführt, nachdem alle Tasks nach Measure Events beendet sind. Das heißt, das, was Sie
369 vorhin eigentlich schon angesprochen haben, man wartet erst alle Zustandsabfragen ab und
370 dann würde erst der Strang in Control weiter verarbeitet werden. Also die Tasks, die nach
371 Control Events modelliert sind, würden dann erst weiter abgearbeitet werden. #00:38:54-8#

372 64. B: Okay, und bei parallel läuft auf der einen Seite schon das Measure und gleichzeitig läuft auch
373 ein Control, oder? #00:39:10-5#

374 65. I: Genau, ja. #00:39:10-9#

375 66. B: Und wie/ nimmt das Control die Daten aus dem vorherigen Measure? Weil wenn das parallel
376 läuft, kann ja das Measure in dem Strang noch nicht fertig sein, wenn das Control schon beginnt.
377 #00:39:24-3#

378 67. I: Genau, es würde dann der letzte Wert übernommen werden. Ja. #00:39:28-3#

379 68. B: Okay. #00:39:30-3#

380 69. I: Für technische Ausführungen, wenn man sich jetzt speziell SPS'en, speicherprogrammierbare
381 Steuerungen, anschaut, dann/ ich weiß nicht, ob Sie damit vertraut sind, aber normalerweise
382 würde man dann ein Prozessabbild machen. Also man hätte ein ganzes Set an Messungen und
383 mit DENEN würde man dann erst in die Verarbeitung hineingehen. Für verschiedene Prozesse
384 wollten wir aber grundsätzlich auch die Möglichkeit geben, dass wir parallel und sequentiell,
385 dass wir diesen Unterschied einfach anbieten für die Modellierung. Also derweil wollten wir
386 einfach beide Möglichkeiten zur Verfügung stellen. #00:40:11-1#

387 70. B: Ja, also ich sehe das schon (hustet) als ganz gut, dass man eben beides kann. Nur was ich/
388 parallel glaube ich, könnte halt verständnisweise ein bisschen schwierig sein, ja, für die Leute.
389 Aber das sehen wir dann eh an den Beispielen dann, ob das so ist oder nicht. #00:40:35-1#

390 71. I: Wobei ich dazu sagen muss, dass wir hier bei den beiden Beispielen aus den Unterlagen in
391 beiden Fällen sequentiell modelliert haben. //B: Ah, okay. // Also wir haben leider kein, in dieser
392 Fassung des Leitfadens, kein Parallelbeispiel drinnen //B: Okay. // aber eventuell muss ich das
393 noch nachholen. #00:40:54-1#

394 72. B: Gut, aber vielleicht habe ich dann noch Fragen zu parallel oder so. Da kann ich ja dann das
395 anführen oder so. Oder du hast es dir eh schon notiert oder, keine Ahnung. #00:41:07-8#

396 73. I: Ja, wir können einfach mal weiter durchgehen. #00:41:12-7#

397 74. B: Ja, ja. #00:41:13-6#

398 75. I: Okay. Die Events, von den Erweiterungen. In einem Closed Loop Subsystem werden spezifische
399 Ereignisse erwartet, die in eine der drei folgenden Kategorien fallen. Ereignisse für
400 Zustandsabfragen oder Messungen. Ereignisse für Regulierungen, oder Regelungen und
401 Ereignisse für die Unterbrechung des Closed Loop Subsystems. Es gibt für jede Ereigniskategorie
402 zumindest eine Kante, die vom Gateway ausgeht. Die Kanten zeigen an, welche Tasks
403 nebeneinander ablaufen. Sobald diese Ereignisse eintreten, werden auch die Tasks, die in den
404 Kanten danach angeordnet sind, ausgeführt. Sie sehen hier gleich ein Bild eines Closed Loop
405 Subsystems, in dem nur Ereignisse der drei Kategorien, ohne darauf folgende Tasks modelliert
406 sind. Das heißt, würde man ein Closed Loop Subsystem einfügen in einem Prozess, würde es von
407 Anfang an so hier aussehen, wie Sie es hier sehen, auf dem Bild. #00:42:17-8#

408 76. B: Und das Cancel sollte aber ganz unten stehen, oder? // I: Das/ // Beim X. #00:42:24-2#

409 77. I: Ach so. Nein, das ist die Variante, wie das Closed Loop Subsystem modelliert wird.
410 #00:42:29-5#

411 78. B: Ach so. Okay. #00:42:31-7#

412 79. I: Also, hier könnte cancel oder wait stehen. In dem Fall //B: Okay, okay. // steht hier cancel.
413 Oder beziehungsweise parallel oder sequentiell. #00:42:40-1#

414 80. B: Gut. Ach so, das war das. Okay, ich weiß schon. Und wieso steht beim X (Symbol für Cancel
415 Event) nichts, was das ist? #00:42:47-4#

416 81. I: Weil hier die Abbruchbedingung noch nicht definiert ist. #00:42:51-2#

417 82. B: Okay. #00:42:52-2#

418 83. I: Also hier würde/ hier geht es im Grunde einfach um eine Bedingung, die man definieren
419 müsste. Die würde auch als Label dann neben diesen Event erscheinen, das heißt, //B: Okay,

420 verstehe. // das kommt eh dann ein bisschen später, dass wir das dann noch definieren, aber das
421 könnte zum Beispiel sein, wenn irgendein/ #00:43:10-5#

422 84. B: Ja, wenn dieser Werte halt zu groß ist, dann wird abgebrochen. #00:43:13-9#

423 85. I: Zum Beispiel, ja. Genau. #00:43:15-8#

424 86. B: Ja. Okay. #00:43:16-8#

425 87. I: Okay. Die drei Ereigniskategorien, die wir definiert haben, sind wie folgt. Measure/
426 #00:43:22-3#

427 88. B: Nur kurz. Was wird abgebrochen? Der ganze Prozess, also diese ganze Closed Loop oder nur
428 dieser eine Durchlauf? //I: Nein/ // Weil es gibt ja zwei Möglichkeiten. #00:43:37-6#

429 89. I: Bei Cancel Events würde das Closed Loop Subsystem abgebrochen werden. #00:43:42-5#

430 90. B: Okay, passt. #00:43:44-1#

431 91. I: Measure Events. Empfängt Events für die Ausführung von Tasks in Messzyklen,
432 Zustandsabfragezyklen. Control Events empfängt Events für die Ausführung von Tasks in
433 Regelzyklen oder Regulierungszyklen. Und Cancel empfängt Events für das Abbrechen von
434 Closed-Loop-Systemen. #00:44:07-2#

435 92. B: Ja. Steht eh da. #00:44:09-6#

436 93. I: Ja, ist ja kein Problem, wenn Sie fragen. Also (lacht), das passt schon. Diese Symbole geben den
437 Zweck der nachfolgenden Tasks oder Aufgaben an. Diese Tasks werden nur ausgeführt, wenn die
438 Ereignisse ausgelöst werden. Das bedeutet, dass das Messereignis angibt, dass die
439 nachfolgenden Symbole nur Messabläufe beziehungsweise Zustandsabfragen anzeigen. Das
440 Gleiche gilt dann für Regelungs- oder Kontroll- und Abbruchereignisse. Für Zustandsabfragen
441 und Regulierungen können wir eine Zykluszeit definieren. Dadurch kann die Dauer von
442 Anpassungen im System definiert werden. Je nachdem, ob das Closed Loop Subsystem einen
443 parallelen oder sequentiellen oder einen Wait- oder Cancel-Ansatz verfolgt, läuft die Ausführung
444 unterschiedlich. Mit diesen Bedingungen kann man definieren, inwiefern Anpassungen beim
445 System erfolgen. Hier sehen Sie ein Closed Loop Subsystem mit einem Task für eine Messung. In
446 diesem Fall wird das Ereignis für die Messung alle zehn Sekunden getriggert. Danach wird der
447 Wert V 1 geholt beziehungsweise gemessen. Wait bedeutet, dass ein neuer Zyklus erst startet,
448 wenn die Messung erfolgt, das heißt, der Prozess in dieser Kante abgeschlossen ist. Cancel
449 würde bedeuten, dass nach zehn Sekunden automatisch der neue Zyklus gestartet wird.
450 #00:45:32-5#

451 94. B: Das mit den zehn Sekunden sehe ich an den Labels nicht, oder? #00:45:39-1#

452 95. I: Das ist hier als Frequenz definiert. #00:45:42-1#

453 96. B: Ah, das ist das. Okay. #00:45:44-3#

454 97. I: Das heißt, eine Frequenz von 0,1 Hz wären einmal alle zehn Sekunden. (...) Der Kehrwert.
455 #00:45:57-8#

456 98. B: Können wir nochmal kurz ein bisschen hinauf scrollen? Nur den Absatz. (unv.) wird das
457 Ergebnis für die Messung alle zehn Sekunden getriggert. Das heißt, ich definiere jetzt nicht die
458 zehn Sekunden für das ganze Closed Loop System, sondern für jeden einzelnen Schritt, oder?
459 #00:46:23-3#

460 99. I: Hm. (zustimmend) #00:46:24-1#

461 100. B: Okay, das habe ich am Anfang falsch verstanden. Ich habe nämlich geglaubt, das ist für das
462 ganze System definiert, dass das zehn Sekunden dauert, ja, und dann werden alle diese
463 einzelnen Kanten, werden alle abgebrochen. Okay, das ist natürlich viel besser, wenn wir das pro
464 Schritt egal ob Zustandsabfrage oder Regulierung, dass man das bei jedem definieren kann. Das,
465 ja, finde ich gut, okay. #00:46:55-2#

466 101. I: Okay. Also grundsätzlich hätten wir gedacht, dass, wenn es sich hier um einen Prozess handelt,
467 mit zusammenhängenden/ also mit einer Verbindung zwischen den jeweiligen Abfragen und
468 dem jeweiligen Kontroll-Strang, dass das sehr wohl eine Auswirkung auf das komplette Closed
469 Loop System hat. Aber es gibt zwei unterschiedliche Bedingungen für die Frequenzen. In dem
470 Fall. Also, wenn sich jetzt die Frequenz von Measure unterscheiden würde von der Frequenz von
471 Control, sagen wir Control hätte eine höhere Frequenz beispielsweise als Measure, dann würde
472 das Closed Loop System auch abbrechen, wenn man cancel definiert hätte und Control im
473 Control-Strang die Tasks nicht rechtzeitig abgearbeitet werden. #00:47:51-3#

474 102. B: Okay. #00:47:54-3#

475 103. I: Das ist dem jeweiligen Modellierer dann überlassen, dass er das dann auch richtig definiert.
476 //B: Okay. // Also natürlich müsste das dann auch zusammen passen, dass Measure Events
477 ähnlich oft oder häufiger abgefragt oder getriggert werden wie Control Events. Weil natürlich die
478 Werte, die vorher, also die Zustandsabfragen, die vorher erfasst wurden, wichtig sind für die
479 entsprechende Regulierung. #00:48:16-9#

480 104. B: Okay, ja. #00:48:22-3#

481 105. I: So zur Erklärung. Was kann man jetzt beim Measure Event definieren? Wie gesagt die
482 Intervalldauer in Hertz, also die Frequenz. Und man kann auch hier definieren in der jeweiligen
483 Kante, also die Tasks, die dann nach dem jeweiligen Measure Event eingefügt werden, welche
484 Werte hier dann geändert werden oder überschrieben werden. (...) Ist es okay, wenn ich
485 weitermache oder? #00:49:02-7#

486 106. B: Ja. #00:49:03-9#

487 107. I: Okay, gut. Mithilfe von Regelungsereignissen kann ferner festgelegt werden, welches
488 Reglermodell verwendet wird, also PID zum Beispiel, PI, PD. Diese Regler werden in ihrer
489 mathematischen Form dargestellt. Die Tasks für sie sind im Grunde Berechnungen, die in festen
490 Teilprozessen dargestellt werden können. Nach diesen Berechnungen kann der Benutzer Tasks
491 zur weiteren Datenverarbeitung hinzufügen. Deswegen könnte man auch sagen, dass das nicht
492 nur Zustandsabfragen oder Mess-Tasks, sondern Datenerfassungs-Tasks sein könnten. Hier
493 sehen Sie jetzt ein Prozessmodell mit einem Wert, der gemessen wird, und einer
494 darauffolgenden Regelung. Die verschiedenen Schritte sind hier all Scripts dargestellt und dann
495 folgt ein Service Call darauf, der dafür stehen soll, dass das entsprechende Signal oder der
496 entsprechende Wert, der auf diesen Berechnungen hier basiert, also auf den jeweiligen
497 Umrechnungen in den Scripts definiert, ausgerechnet wird und DER dann an den jeweiligen
498 Aktor geschickt wird. Das heißt, wir haben einmal in der ersten Kante die Abfrage von V 1, dann
499 haben wir die Differenzberechnung beispielsweise von V opt und V 1, also V 1 wird von V
500 optimal abgezogen. Dann, mit dieser Differenz könnten wir in den PID-Code hineingehen,
501 könnten uns unser jeweiliges Ergebnis ausrechnen und dieses dann an den jeweiligen Aktor
502 schicken. Ganz einfach dargestellt. Entsprechende Datenelemente könnte man auch in der [REDACTED]
503 einfügen. Sie wissen eh, wie man das grundsätzlich verwendet //B: Ja. // oder wofür man sie
504 einsetzen kann. Bei Control kann man die folgenden Attribute definieren, auch wieder wie
505 gesagt Interval frequency in Hertz. Wäre in diesem Fall gleich wie bei Measure. Und dann kann
506 man noch die Values expected to change auch angeben. Man kann hier aber ein paar mehr
507 Daten angeben, als bei Measure. Nämlich welchen Controller Type man verwenden möchte, also
508 in unserem Fall in diesem Beispiel wäre das jetzt PID-Regler. Der Wert, der sich ändert. Das
509 Upper Limit und das Lower Limit. Das heißt in welcher Range dürfen wir uns damit bewegen?

510 Das soll hier dazu beitragen, dass, wenn jetzt irgendetwas mit dem System nicht stimmen würde
511 oder entsprechend falsche Werte vielleicht herauskommen bei der Berechnung und, sagen wir
512 zum Beispiel es geht um eine einfache Temperatur Regelung. Und das System sagt, 'Das Medium
513 ist so kalt, wir müssen jetzt hoch heizen. Wir müssen mit einer höheren Heizleistung
514 hineingehen.' Dann kann man hier aber eine obere Grenze einziehen, die in diesem System,
515 wenn jetzt kein Fehler vorliegt, also wenn es normal funktionieren würde, nicht überschritten
516 werden dürfte. #00:52:17-8#

517 108. B: Okay. Eben, um eben falsche Messungen //I: Genau. // auszutarieren, damit man nicht, keine
518 Ahnung, misst das Ding halt null Grad, derweil hat es aber eh fünfzig Grad und man heizt
519 nochmal ordentlich hinauf und dann/ Okay. #00:52:32-8#

520 109. I: Genau, es ist eine Sicherheitsmaßnahme, kann man sagen. Ja. Wait bedeutet wieder, dass für
521 den nächsten Zyklus auf das Beenden aller Tasks gewartet wird, auch auf die Regulierungstasks.
522 Sequentiell heißt, dass die Tasks nacheinander ausgeführt werden. Das heißt, es wird erst
523 gemessen beziehungsweise der Zustand abgefragt vom System und mit diesem gemessenen Wert
524 wird die Regelung durchgeführt. Wie gesagt, wird die Differenz erst ausgerechnet und auf Basis
525 dieser Differenz kann man dann den jeweiligen Wert ausrechnen, den man dann auch weiter an
526 den Aktor schicken könnte. Wie gesagt, über ein Service Request hier dargestellt. Das würde an
527 das jeweilige Stellglied geschickt werden. Das wäre ein Element, das aktiv Einfluss auf den
528 Prozess ausübt. Und wenn man möchte, kann man die Differenzberechnung, die
529 Regelungsberechnung, PID Code in dem Fall hier als Beispiel dargestellt, und das Aussenden des
530 Befehls an das System in einen Subprozess zusammenfassen. Bei Control können zusätzlich die
531 Art der Regelung, wie gesagt, sowie der Stellwert und dessen Limits eingetragen werden. Würde
532 hier parallel verwendet werden, würde der letzte Wert von V 1 genommen werden. Das haben
533 wir eh vorhin besprochen, das Beispiel. Also der Unterschied zwischen parallel und sequentiell.
534 Zustandsabfragen und Regulierungen sollten in regelmäßiger Frequenz ausgelöst werden,
535 Abbruchereignisse natürlich nicht. Diese werden hingegen nur durch ihre Abbruchbedingungen
536 ausgelöst, die der Benutzer definieren kann. Ein Beispiel für ein Abbruchereignis wäre, wenn
537 etwas den Abbruch eines Zyklus auslöst, also grundsätzlich den Abbruch des kompletten
538 Subsystems. Das könnte beispielsweise ein Notstoppsignal sein, das ausgelöst wird. Oder wie Sie
539 vorhin auch schon erwähnt haben, als Beispiel wenn der Kessel leer ist oder wenn sonstige
540 Zustände dazu führen würden, dass man den Prozess beenden sollte. #00:54:47-8#

541 110. B: Ja. #00:54:50-3#

542 111. I: Hier in dem Fall haben wir unser voriges Beispiel genommen und haben es um eine
543 Abbruchbedingung erweitert. Das wäre hier Emergency Stop active, wenn der auf true gesetzt
544 wird. Das heißt, wenn der Notstopp aktiviert wird, dann müsste natürlich das Closed Loop
545 System beendet werden. Weil dann haben wir keinen kontinuierlichen Prozess mehr, dann
546 haben wir eine Abbruchbedingung. //B: Okay. // Standardmäßig würde der Wert aber bei
547 Default auf false sein. Das heißt, wir gehen natürlich davon aus, dass der nicht von Anfang an auf
548 true gesetzt wird, das nicht von Anfang an einen Notstopp aktiviert ist. #00:55:29-0#

549 112. B: Okay. Ja, okay. #00:55:38-0#

550 113. I: Sobald die Abbruchbedingung, Emergency Stop active hier, true wird, werden repetitive Tasks
551 beendet. Die Abbruchbedingungen werden grundsätzlich bei jedem Zyklus neu evaluiert, solange
552 sie natürlich nicht ausgelöst werden. Weil, dann würde man natürlich aus dem kompletten
553 System ausbrechen. //B: Ja. // Nachdem das Ereignis ausgelöst wurde, können Tasks zur
554 Aufräumaroutine abgearbeitet werden, wenn man diese definieren möchte, bevor der komplette
555 Prozess vollständig beendet wird. #00:56:12-8#

556 114. B: Darf ich noch etwas sagen? //I: Ja. // Ich glaube, Emergency Stop ist ein schlechtes Beispiel.
557 #00:56:20-7#

558 115. I: Okay. Was würden Sie hier einfügen? Oder warum meinen Sie, wäre das ein schlechtes
559 Beispiel? #00:56:27-7#

560 116. B: Weil beim Emergency Stop darfst du auch keine/ Du darfst dann auch keine Aufräumarbeiten
561 mehr durchführen, weil Emergency Stop heißt, Ende. Die Maschine wird heruntergefahren, ja.
562 Das ist zum Beispiel, wenn ein Mitarbeiter beim Roboter herumläuft, ja, und der sollte dort nicht
563 sein, ja, dann wird Emergency Stop ausgeführt. Der Roboter wird gestoppt, ja. Der macht keine
564 Aufräumarbeiten mehr, dass er sich in die Default-Position bewegt, ja. Weil genau bei diesem
565 Ding könnte ja diesen, den Mitarbeiter, der dort herumläuft, verletzen, ja. Also, ich glaube
566 Emergency Stop ist nicht das beste Beispiel, zumindest wenn man sagt, man macht danach noch
567 Aufräumarbeiten. #00:57:20-5#

568 117. I: Ja, ich verstehe, was Sie meinen. Aber es könnte zum Beispiel auch sein, dass es sich hier
569 einfach nur um eine Teilroboterzelle handelt, nur um einen gewissen Arbeitsbereich, also man
570 kann auch den Notstopp für einen gewissen Arbeitsbereich machen und kann dann aber, wenn
571 der Notstopp erfolgt, wirklich noch etwas, vielleicht ein Signal an eine andere Zelle schicken, für,
572 keine Ahnung, für die Benachrichtigung eines gewissen Zustands oder dergleichen. #00:57:45-3#

573 118. B: Ja, natürlich. Es gibt Szenarien, wo das dann durchaus okay ist, ja. //I: Ja. // Dass man dann
574 auch bei einem Emergency Stop in einen Zustand kommt. Zum Beispiel. Wenn ein Emergency
575 Stop bei einem Atomreaktor ist, dann wird trotzdem weiter gekühlt, weil sonst geht das Ding in
576 die Luft, ja. Also, natürlich kann ein Emergency Stop auch zu weiteren Prozessschritten führen. Ja,
577 also das schon. Es ist nur vielleicht auch ein kritisches Beispiel, also. #00:58:18-4#

578 119. I: Das stimmt, ja. Es regt zur Diskussion an, ja. #00:58:20-7#

579 120. B: Aber es ist verständlich, ja, also. #00:58:26-9#

580 121. I: Ja, als besseres Beispiel, oder einfacheres, könnte man hier zum Beispiel, wie Sie vorhin schon
581 genannt haben, einfach den Füllstand des Kessels überwachen. #00:58:38-7#

582 122. B: Ja, zum Beispiel. Also dann gehe ich dann auch in einen konsistenten Zustand und fertig. Und
583 diese Abfrage der Bedingung, ja, erfolgt die immer ganz am Anfang, wenn die Loop durchläuft
584 oder erfolgt die auch zum Beispiel, weiß ich nicht, es läuft alles durch, es kommt ein neuer
585 Measure. Kann dieser neue Measure auch sofort zu dieser Bedingungen führen oder erst beim
586 nächsten Durchlauf? #00:59:15-8#

587 123. I: Wir haben im zweiten der Beispiele, die wir modelliert haben, die dann gleich auch kommen,
588 die Variante, dass ein Zustand abgefragt wird und ein gewisser Wert dieses Zustands dann dazu
589 führt, dass das komplette System halt abgebrochen wird. #00:59:41-0#

590 124. B: Okay, das heißt, sofort. Das heißt, ich muss nicht erst diesen Zyklus durchmachen und dann
591 kommt es zum Abbruch. #00:59:47-9#

592 125. I: Ja, also das soll nämlich auch den Sinn eigentlich erhalten, dass wenn der Abbruch getriggert
593 wird, dass dann entsprechend auch nicht mehr reguliert werden darf. #00:59:58-8#

594 126. B: Genau. Okay, passt. #01:00:02-7#

595 127. I: Okay, das habe ich bereits erwähnt. Genau, und dann hätten wir halt nach einem besseren
596 Beispiel, als Emergency Stop beispielsweise, //B: Ja, ja. Das passt schon. (unv.) // also
597 Füllstandsüberwachung, okay, haben wir dann halt noch eine Aufräumroutine definiert. Also hier
598 in Form eines Service Calls dargestellt, wenn wir das machen möchten. Die vorgestellten
599 Erweiterungen sollen bei der Modellierung von kontinuierlichen Prozessen helfen, indem
600 Vorlagen für die Erstellung von Prozessmodellen vorgegeben werden und andererseits durch die
601 Darstellung als Closed Loop Subsystem mit eigenen Symbolen für Zustandsabfrage-,
602 Regulierungs- und Abbruchereignisse helfen, solche Prozesse leichter nachvollziehen zu können.

603 Hinzu kommt, dass man für eine übersichtlichere Darstellung des gesamten Prozesses auch
604 Subprozesse zur Unterteilung nutzen kann. Damit erfüllen wir auch das letzte Feature, dass die
605 Verständlichkeit der Modelle von kontinuierlichen Prozessen auch gegeben werden soll, oder
606 gegeben sein soll. So. Zu den Beispielen. Ich werde Ihnen nun Prozessbeispiele zeigen, die mit
607 denen unserer Arbeit vorgestellten Erweiterungen modelliert sind. Ich möchte, dass Sie sich die
608 Modelle ansehen und mir sagen, was Sie aus Ihnen herauslesen können und ob die Modelle den
609 notwendigen Informationsgehalt für die Modellierung der zugrundeliegenden
610 Regelungsprozesse erfüllen. Vorab wird Ihnen zum jeweiligen Prozess erklärt, was auch
611 abgebildet werden soll. Und ich würde Sie bitten, offenes Feedback zu den Modellen zu geben.
612 Ich mache hier ganz kurz eine Pause. #01:01:52-4#

613 128. B: Ja. #01:01:53-5#

614 UNTERBRECHUNG - 5 Minuten

615 129. I: Okay, Aufnahme läuft wieder. Wir haben bei der Intro (Einführung) für die Prozessmodelle
616 aufgehört. Und zwar. Kommen wir mal zum ersten Beispiel. Es handelt sich hier/ #00:00:13-9#


617 130. B: Moment, ich mache nur schnell da die Tür zu. //I: Ja. // Weil ich höre da die ganze Zeit etwas.
618 #00:00:19-4#

619 131. I: Okay. #00:00:19-8#

620 132. B: So, jetzt. Geht schon. #00:00:23-2#

621 133. I: Also, das erste Beispiel ist eine einfache PI-Temperaturregelung mit einem Wärmetauscher
622 basierend auf dem Beispiel aus der MathWorks-Bibliothek, also MATLAB. Die Temperatur einer
623 Flüssigkeit in einem Rührkessel wird mittels Wärmetauscher geregelt. Der über den
624 Wärmetauscher eingebrachte Wärmestrom wird über ein Ventil, das den Dampfstrom, also den
625 eigentlichen Wärmeträger, kontrolliert, gesteuert. Der zu beachtende störende
626 Umgebungseinfluss ist die schwankende Temperatur der zugeführten Flüssigkeit, die von oben
627 in den Kessel eingeführt wird. Der Tank wird als isoliert angenommen. Das heißt, wir gehen nicht
628 davon aus, dass über den Mantel des Tanks irgendwelche/ #00:01:15-6#

629 134. B: Wärmeaustausch erfolgt. Okay. #00:01:17-7#

630 135. I: Genau, ja. Das Flowchart des Prozesses würde so aussehen. Man sieht hier den
631 Wärmetauscher, den Tank mit dem entsprechenden Rührwerkzeug, den Temperaturfühler, den
632 Inflow von oben und natürlich hier auch links das Ventil mit der entsprechenden Steuerung.
633 Dann haben wir einige Datenelemente, auch für die Temperaturregelung, also für das
634 Reglermodell definiert, die wir später brauchen. Das kennen Sie eh, wie man das in der 
635 definiert, wie man auch die Endpunkte definiert, die man benutzt. Und dann haben wir die
636 Attribute, die wir für unser Closed Loop System definiert haben. Wir haben hier wait und
637 sequential. Wobei über die MathWorks-Bibliothek keine zeitlichen Vorgaben bestehen. Das
638 heißt, man könnte hier eigentlich auch parallel nehmen. Also wir haben hier in den Unterlagen
639 keine konkreten Vorgaben, wie die zeitlichen Constraints ausschauen. Wir messen erst einmal
640 die Temperatur des Tanks. Wir messen anschließend dann die Temperatur der Disturbance, also
641 der Störgröße. Das wäre der Zufluss von oben. Dann haben wir im weiteren Strang das Control
642 Event mit dem jeweiligen Control-Modell, also PI-Modell, in dem Fall. Dann könnten wir noch
643 eine Umrechnung einfügen, wenn wir es für notwendig halten. Und dann wird der
644 entsprechende Wert, den wir wieder heraus bekommen, natürlich an den Aktor geschickt. Die
645 Abbruchbedingung haben wir hier in MathWorks ebenfalls nicht definiert, vorgegeben.
646 Deswegen ist einfach wieder eine beliebige Abbruchbedingung hier modelliert. In dem Fall

647 haben wir sie einfach Stop activated genannt. Das heißt, wenn der Stopp aktiviert wird, dann
648 wird eine entsprechende Shutdown-Sequenz ausgeführt. Die ist jetzt hier als Script dargestellt,
649 aber natürlich könnte das ein Subprozess sein, oder eine weitere Reihe von verschiedenen Tasks,
650 die man hier hinein modellieren möchte. Dann sehen Sie noch das mathematische Modell für
651 den PI-Controller, eine Reihe von mathematischen Operationen, Umwandlungen. Und ja, ich
652 würde Sie bitten, das Modell nach folgenden Kriterien auf einer Skala von Eins bis Fünf zu
653 bewerten, wobei Eins sehr schlecht ist und Fünf, sehr gut. Und zwar geht es mir um die
654 Verständlichkeit. Das heißt, können Sie sagen, was hier passiert, aufgrund des Modells?
655 Übersichtlichkeit. Können Sie das gesamte System auf einen Blick erfassen? Einfachheit. Könnte
656 man das Modell vielleicht noch einfacher darstellen? Die Logik. Wird klar, was parallel und was
657 sequentiell passiert? Und die Erweiterbarkeit. Das heißt, könnte man dem Modell noch etwas
658 hinzufügen, was den Informationsgehalt verbessern würde? #00:04:13-5#

659 136. B: Okay. Ist es okay, wenn ich mir das Modell parallel dazu aufmache? #00:04:19-8#

660 137. I: Ja, sicher, natürlich. Bitte. #00:04:21-7#

661 138. B: Passt. Gut. Verständlichkeit. Es wäre ein/ Also zur Verständlichkeit ist natürlich zu sagen, dass/
662 ich habe das Wait. Das heißt, ich warte immer auf die einzelnen Stränge, bis die fertig sind. Und
663 das Ganze läuft sequentiell ab, wenn ich das richtig verstanden habe. Mit sequentiell ist, dass
664 zuerst das Measure durchgeführt wird, dann das andere Measure. Die glaube ich können aber
665 parallel auch stattfinden, diese Measures, oder? #00:04:59-0#

666 139. I: Genau. Wenn die Frequenz die gleiche wäre, können die parallel ohne Probleme stattfinden.
667 Also das wäre der Sinn dahinter, ja. #00:05:06-3#

668 140. B: Genau. Und wenn das sequentiell ist, werden die Measures durchgeführt und danach wird das
669 Control ausgeführt, mit eben den entsprechenden Aktivitäten ebenfalls. #00:05:18-8#

670 141. I: Genau. #00:05:19-6#

671 142. B: Gut, das Stop ist, ja, sowieso klar, wann das ausgeführt wird, wenn irgendeine Bedingung
672 erfüllt ist. Gut also, ich würde sagen, also, das ist für mich schon sehr verständlich. Auch dass
673 man eben bei Get tank, da kriegt man, also, dass man dort die verschiedenen Temperaturen
674 misst. Das ist soweit klar und auch eben, was ich hier mittlerweile verstanden habe, ist, dass
675 eben bei den Measures die einzelnen Wartezeiten angeführt werden. Ist mir auch mittlerweile
676 klar, wie lange das dauert. Also ja, das ist für mich eigentlich alles sehr gut verständlich. Aber nur,
677 muss ich schon auch sagen, durch die intensive Einführung, ja. Also, DIE ist natürlich definitiv
678 Voraussetzung für das Ganze. #00:06:16-9#

679 143. I: Verstehe, ja. #00:06:19-7#

680 144. B: Also. Ob es jetzt intuitiv ist, kann ich jetzt schwer bewerten, ja. Weil es ja jetzt diese
681 Einführung gegeben hat. Ich finde es jetzt soweit intuitiv, dass ich es nach dieser doch sehr
682 kurzen Einführung glaube ich recht gut verstehe, was passiert. Also von der Verständlichkeit her
683 würde ich es schon als sehr gut bewerten, weil mir klar ist, was wann passiert. #00:06:51-2#

684 145. I: Okay. Dankeschön. Was würden Sie zur Übersichtlichkeit sagen? #00:07:02-5#

685 146. B: Bei dem Beispiel finde ich es NOCH ganz gut, weil es noch nicht/ Es ist halt kompakt, ja. Also
686 es sind nicht viele Aktivitäten, es sind drei Measures. Das ist okay, also das ist nicht aufgebläht.
687 Da gibt es sicher komplexere Beispiele. Also. Übersichtlich finde/ was halt schon ein bisschen die
688 Übersichtlichkeit trübt, ja, ist vielleicht diese lange Liste an Labels auf der rechten Seite, ja. Das
689 war schon fast wie ein eigener Paragraph, der wirkt, ja. Also beispielsweise steht da immer
690 Measure Doppelpunkt, ja. Wenn man ja die Symbole erkennt/ ich meine, ich verstehe es, (unv.,
691 Dual Coding?) ist eigentlich schon etwas, das was man anwendet, ja. Aber an und für sich sagt ja
692 schon das Symbol, dass es Measure ist, ja. //I: Ja. // Das ist die Frage, was man da möchte. Also

693 ich finde es auf der einen Seite natürlich gut, dass das dabei steht. Weil wenn man sich jetzt
694 nicht ganz sicher ist, dann kann man nochmal/ sieht man es nochmal auf der rechten Seite, dass
695 da eben das Measure passiert, ja. Auf der anderen Seite macht es halt diese Labels nochmal
696 länger, ja. #00:08:34-0#

697 147. I: Ja. #00:08:34-5#

698 148. B: Die Übersichtlichkeit jetzt vom ersten mit diesem Wait und Sequential, das finde ich super, ja.
699 Das sind zwei Keywords und die sind essentiell, ja. Weil die sind/ die können ja da immer anders
700 sein, weil statt dem Wait kann ein Cancel sein und statt dem Sequential kann ein Parallel sein.
701 Das finde ich gut. Das finde ich auch sehr kompakt. Das würde ich vielleicht auch nicht in das
702 Symbol mit hineinpacken, ja. Das glaube ich, könnte das Symbol zu komplex machen, wenn man
703 jetzt ein Symbolkombination macht, ja. Dass man auf der linken Seite entweder dieses Wait oder
704 Parallel da hinein zeichnet, und auf der rechten Seite vom Symbol das Sequential/ Wait oder
705 Cancel, und auf der rechten Seite das Sequential oder Parallel. Würde ich jetzt vielleicht auch gar
706 nicht in das Symbol mit hinein packen. Wäre ja auch eine Möglichkeit, würde ich aber vielleicht
707 jetzt nicht machen. //I: Okay. // Weil es dann einfach, ja, ausartet in zu viele Symbole. Gut, was
708 haben wir noch? Ich bin gerade verwundert, dass ich da auf gemutet (stumm geschalten)
709 angezeigt werde. #00:09:46-0#

710 149. I: Okay, ich höre Sie aber ohne Probleme. #00:09:48-9#

711 150. B: Okay, na passt. Gut. Also, Übersichtlichkeit. Da schwanke ich so ein bisschen zwischen/ ja. Ich
712 würde dem eine Vier geben, weil ich denke, da ist noch Potenzial nach oben. #00:10:09-8#

713 151. I: Okay, ja, verstehe. Nächster Punkt wäre Einfachheit. Könnte man das Modell noch einfacher
714 darstellen? #00:10:20-6#

715 152. B: Was mir da halt einfallen würde, ist die Frage, ob man zum Beispiel bei Control, da wirklich
716 drei aktive/ Das ist halt in dem Sinne, ja, ist halt die Frage, ob ich da jetzt das Prozessmodell
717 immer (unv., angewendet?) beschreibe, oder dieses kontinuierliche Modell, das System, also die
718 eingeführten Symbole, die eingeführte Logik, ja. Weil ansonsten könnte man dann überlegen, ob
719 man die wirklich drei Aktivitäten unter Control braucht, ja. #00:11:03-9#

720 153. I: Ja. #00:11:05-2#

721 154. B: Das ist/ oder ob man das eh nicht in nur eine Aktivität zusammenfassen könnte. Das ist aber
722 der eigene Modellierungsstil. Also, das würde ich jetzt der Stelle so nicht bewerten, ja, sondern
723 würde bei Einfachheit bewerten, dass es eben gibt, dieses Loop-Symbol mit dem momentan vier
724 Strängen, ja, und das sehe ich als sehr einfach an. Das ist sehr/ Ich glaube, das ist/ Das kann man
725 gar nicht simpler darstellen, ja. Also. #00:11:40-1#

726 155. I: Okay. #00:11:41-4#

727 156. B: Also, das Einordnen würde ich mit sehr gut bewerten. #00:11:49-9#

728 157. I: Okay, danke. Der vorletzte Punkt wäre Logik. Wird klar, was parallel und was sequentiell
729 passiert? #00:11:58-7#

730 158. B: Ja, also. Ob es/ Ganz, ganz klar würde ich sagen, ist es vielleicht nicht. Weil ich habe ja auch
731 nachfragen müssen jetzt, ob diese Measures nacheinander ablaufen oder parallel, ja. //I: Genau.
732 // Also als ganz, ganz klar sehe ich es nicht. Ja, das/ Ich wüsste da jetzt so auf die Schnelle nicht,
733 wie man das besser machen könnte, ja. Eine Möglichkeit wäre zum Beispiel, dass man die
734 Measures, wenn sie parallel ablaufen, auf die gleiche Höhe zieht, ja. Dann hat man aber wieder
735 das Problem, dass man diese Labels so nicht einfügen kann, ja. Deswegen, also, als ganz klar
736 erachte ich es jetzt für mich persönlich nicht, wie der Ablauf ist. Also, mittlerweile weiß ich es,
737 dass die Measures da parallel ablaufen und bei sequential dann das Control kommt. Aber. Das

738 hängt ja dann nochmal an der Frequenz, mit der Frequenz zusammen, //I: Genau. // also das
739 finde ich jetzt nicht als ganz einfach. Da würde ich jetzt von dem Ganzen, würde ich jetzt weder
740 noch nehmen. Also es ist jetzt nicht super komplex, ja. Ich glaube, wenn man sich da eine Stunde
741 damit einarbeitet, dann ist das schon klar. Aber es ist halt auch nicht super simpel, ja. Dass ich
742 jetzt hinschaue und jetzt sofort weiß, was ist sequentiell, was ist parallel. Weil dann muss ich
743 schon zumindest das Modell ein bisschen verstehen vielleicht, ja. Also ich muss da oben noch
744 einmal lesen, aha, da oben ist einmal ein Sequentiell und dann gibt es da noch diese Measures
745 mit dieser Frequenz. Aha, und dann kommt das Control, also, ja. Ich würde es als weder noch
746 einordnen. #00:14:02-0#

747 159. I: Okay, danke. Und der letzte Punkt wäre Erweiterbarkeit. Könnte man Ihrer Meinung nach dem
748 Modell noch etwas hinzufügen, was den Informationsgehalt verbessern würde? #00:14:16-2#

749 160. B: Okay. (...) Also, an und für sich würde ich jetzt nicht großartig etwas hinzufügen, ja, außer
750 vielleicht mit dem, was ich vorhin erwähnt habe bei dem Kesselbeispiel, ja. Dass man halt weiß,
751 die logische Abfolge, in welcher als im echten Prozess diese Measures durchgeführt werden, ja.
752 Das ist da jetzt ein bisschen schwierig, ja, weil das hier in diesem Tank passiert, ja. Aber dass man
753 zum Beispiel eben den Measure hat für Kessel Eins, den Measure für Kessel Zwei. Dass die
754 vielleicht einfach in die Lanes nacheinander kommen, ja. Aber das ist dann auch würde ich sagen
755 Sache des Modellierers. Dass er das genau so anordnet, ja, dass es eben der Realität entspricht,
756 ja. Also der logischen Abfolge, wie es in der Realität passiert, ja. Das würde ich jetzt nicht als
757 Erweiterung hinzufügen, aber vielleicht als Best Practice, wie man so etwas modellieren sollte.
758 Dass man das drinnen hat, weil wenn man schon so klare Schritte hat, die nacheinander
759 passieren, mit den Measures, ja, zumindest real gesehen, obwohl die Measures alle parallel
760 stattfinden, wäre vielleicht so ein Best Practice ganz cool. Ansonsten, ja, ich wüsste jetzt nichts,
761 was man da jetzt noch dazu nehmen sollte. #00:16:18-4#

762 161. I: Okay. Gut, dann können wir schon zum zweiten Beispiel kommen. Im Grunde geht es wieder
763 um eine Temperaturregelung, nur ein bisschen komplexer. Das Modell basiert auf der
764 Beschreibung des Heizprozesses, entnommen aus Schulungsunterlagen der Firma Siemens. Es
765 handelt sich hierbei wie gesagt, ebenfalls um eine Temperaturregelung für einen Rührreaktor.
766 Die Regelung wird in diesem Beispiel mit einem PID-Regler, einer Handsteuerung sowie einem
767 Pulsgenerator realisiert. Die Heizung erfolgt nicht über einen Wärmetauscher, sondern in diesem
768 Fall über ein Heizelement. Weiters sind Verriegelungsbedingungen definiert. Als Basis für die
769 Prozessmodellierung wurden die Beschreibungen wie gesagt aus den Schulungsunterlagen für
770 die Prozessmodellierung mit Simatic PCS 7 von Siemens herangezogen. Unser Prozessmodell
771 wird mit einer automatischen Steuerung modelliert, die mit Umschalten auf Handsteuerung aus
772 dem Closed Loop System ausbricht. Wir gehen davon aus, dass das System bereits angelaufen ist
773 und automatisch gesteuert wird. Weiters wird der Prozess für einen Reaktor, also nur einen
774 einzelnen, und nicht wie in den Unterlagen für zwei Reaktoren beschrieben. Das heißt, wir
775 haben wieder verschiedene Datenelemente, in dem Fall haben wir aber aufgrund der
776 Verriegelungsbedingungen, die definiert sind, auch ein paar Grenzwerte hier drinnen. Wir haben
777 zum Beispiel eine maximale Temperatur vorgegeben für das System, das sind hier sechzig Grad.
778 Wir haben aber auch einen Mindestfüllstand im Reaktor, nämlich 200 Milliliter vorgegeben.
779 Weiters gehen wir davon aus, dass der Betriebsmodus in Automatik läuft und dass der
780 Hauptschalter auf on ist. Das hier ist, ja, ein bisschen komplexer, könnte man sagen. Ich werde
781 kurz heraus scrollen, dass Sie das/ heraus zoomen, dass Sie es vielleicht besser erkennen können.
782 Ist das noch lesbar für Sie? #00:18:31-0#

783 162. B: Ich habe es/ Ja. Ich habe es mir aber eh schon aufgemacht. Weil beim Stream ist es schwer
784 lesbar. #00:18:36-4#

785 163. I: Okay, ja, dann ist es eh besser, wenn Sie es parallel bei sich anschauen können. Wir haben
786 hier/ #00:18:43-1#

787 164. B: Aber es geht, also. #00:18:44-5#

788 165. I: Okay, gut. Also Sie können erkennen, worum es grundsätzlich hier geht? #00:18:47-6#

789 166. B: Ja. #00:18:48-2#

790 167. I: Passt. Wir haben hier fünf Measures, also fünf Stränge mit Measures. Wir natürlich messen
791 einerseits wieder die Temperatur im Reaktor. Wir messen den Füllstand im Reaktor. Wir
792 überprüfen den Operationsmodus. Wieder Emergency Stop und den Hauptschalter, könnte man
793sagen. Dann haben wir ein Control, in dem Fall mit einem PID-Regler. Pulsgenerator und dann
794entsprechend wieder einen Service Call für das Ausschicken des Befehls. Und die
795Abbruchbedingungen sind wie folgt, einerseits, dass der Hauptschalter auf Aus geht. Dass der
796Emergency Stop wieder auf true gesetzt wird. Aber auch in dem Fall, und da haben wir ein
797Beispiel, das Sie vorhin schon angesprochen haben, was man verwenden könnte, nämlich die
798Temperatur des Reaktors, wenn diese größer ist als die Maximaltemperatur, muss das System
799heruntergefahren werden. Aber auch wenn der Füllstand des Reaktors unter dem
800Mindestfüllstand landet. Beziehungsweise, wir verlassen auch dieses Closed Loop System, wenn
801der Operationsmodus auf manuell wechselt. Und wir haben wieder in einem Script, wie oben
802hier dargestellt, das mathematische Modell des Reglers. Ich würde Sie wie vorhin bitten, dass Sie
803das Modell wieder mit den gleichen Kriterien bewerten, nämlich Verständlichkeit,
804Übersichtlichkeit, Einfachheit, Logik und Erweiterbarkeit. #00:20:30-1#

805 168. B: Also, gehen wir einmal zur Verständlichkeit. Ja, es sind halt/ aber das kommt dann eh bei
806Übersichtlichkeit und quasi Einfachheit. Was passiert. Was ich zur Verständlichkeit da in DEM
807Fall ganz gut finde, sind diese, wie soll ich sagen, sprechenden Variablen, ja. Also, Level reactor in.
808Da weiß ich halt genau, was gemessen wird, ja. Damit weiß ich halt auch, was genau bei diesem
809Measure passiert. Also das finde ich ganz gut, aber entsprechend muss man halt dann auch
810modellieren, ja, oder dieses Get operation mode macht es ganz klar, was da passiert. Ich weiß
811jetzt nicht genau, warum ich den Operation Mode getten (engl. get, holen) soll. Ich dachte, das
812funktioniert bei der Abbruchbedingung sowieso automatisch, dass man das dort überprüft.
813#00:21:47-5#

814 169. I: Das haben wir uns jetzt überlegt, dass wir/ Wir sind noch nicht ganz sicher, wie wir das machen
815sollen. Weil bei der Condition könnte man sagen, wir überprüfen eine Variable, die
816überschrieben wird. //B: Genau. // oder wir könnten sagen, es ist, was in einem anderen
817Interview bereits angesprochen wurde, dass das eine Art Push-Nachricht ist, oder/ Wissen Sie,
818was ich meine? Dass das/ Genau, dass man den Wert per Push Notification bekommt, für den
819Prozess. #00:22:17-5#

820 170. B: Genau, ich dachte nämlich eigentlich, vor allem bei so einem wichtigen Ding wie ein Notstopp,
821dass es eben per Push funktioniert, ja. Fände ich jetzt dahingehend einfach/ ich sage das jetzt
822einfach einmal. Ich kann es ja danach dann bewerten, ja, weil ja du dann das Modell ein bisschen
823entschlankst dadurch, ja. Weil du hast dann immer diesen einen Strang für dieses Get operation
824mode oder was auch immer, sondern nur noch unten dieses Cancel Event, ja, wo du dann diese
825Nachricht eben empfangst, ja. Deswegen glaube ich, finde ich das würde es halt ein bisschen
826schlanker machen. Würde aber natürlich zu mehr Logik weiter unten, also im Cancel dann
827führen, ja. Das muss man sich halt überlegen. Sonst hat man ganz klar die Kapselungen des
828Datenelements. Das hat man dann vor allem auch in der Process Engine zugänglich, ja, und kann
829es halt dann auch immer gescheit auswerten, ja. Also das ist halt so ein bisschen die Frage, auch
830auf was man abzielt, was man genau möchte. Gut, also Verständlichkeit. Was passiert ist soweit,
831glaube ich, klar. Nur ich denke, es geht halt alles ein bisschen Hand in Hand mit Verständlichkeit,
832Übersichtlichkeit, Einfachheit. Das sind Sachen, die schließen sich jetzt nicht aus, sondern
833müssen meiner Meinung nach immer miteinander betrachtet werden, ja, und ich denke ein
834bisschen die Verständlichkeit leidet in dem Beispiel an der Komplexität, ja. //I: Ja. // Weil das
835doch sehr, sehr aufgebläht ist, das Ganze, durch diese verschiedenen Stränge. Das macht es jetzt
836nicht unbedingt leichter verständlich, ja. Man muss schon sich ein bisschen mehr einarbeiten.
837Deswegen würde ich jetzt eher sagen, dass/ Das Modell selbst ist denke ich gut verständlich.
838Sehr gut verständlich hingegen würde ich sagen, ist die Notation, ja. Also. Ich weiß nicht, ob das

839 hilft, ja. Weil das Modell selbst ist halt wahnsinnig aufgebläht, aber die Notation SELBST ist mir
840 sehr klar. Jetzt weiß ich nicht, was ich bewerten soll. #00:25:01-8#

841 171. I: Es ginge grundsätzlich um das Modell, aber natürlich, wenn man jetzt/ Sie sind ja auch
842 vertraut mit BPMN. Sie haben ja auch verschiedene Prozesse im BPMN bereits modelliert. Wenn
843 man das jetzt vergleichen würde, wenn man so einen Prozess in BPMN versucht darzustellen, ob
844 die Notation dazu beiträgt, dass man diesen Prozess übersichtlicher oder verständlicher
845 darstellen kann. Also darauf zielen ein bisschen diese Fragen ab. #00:25:30-0#

846 172. B: Hm. (zustimmend) Ja, also da würde jetzt bei Gut bleiben. Ich denke, es ist denke ich schon
847 besser. Aber ich bräuchte jetzt ein bisschen Zeit, dass ich mir den komplett ohne diese Logik da
848 vorstellen könnte, diesen Prozess, ja. Also weil das halt dann doch mit einigen Loops/ und
849 wahrscheinlich ginge es auch gar nicht so, außer mit eben viel Information, die dann gar nicht
850 mehr im Modell ersichtlich ist, ja. Also deswegen/ Ich denke, das ist schon recht gut gelöst. Ja.
851 #00:26:08-8#

852 173. I: Okay. Also, Sie haben damit jetzt eigentlich schon die ersten drei Punkte, oder die ersten drei
853 Kriterien zusammengefasst, könnte man sagen. #00:26:18-9#

854 174. B: Hm. (widersprechend) Also wie gesagt, bei Verständlichkeit bin ich bei Gut. //I: Ja. // Bei
855 Übersichtlichkeit. Da würde ich vielleicht auch bei Gut bleiben. Also ich/ Was ich halt schon auf
856 einen Blick erfassen kann, nach wie vor, sind, dass diese Measures alle im Prinzip zum Beispiel zu
857 nur einem Control führen, ja. Das kann ich/ sehe ich eigentlich auf den ersten Blick, ja. Das sind
858 die Symbole. Da brauche ich nicht einmal die Labels lesen, ist mir das soweit klar, ja. Also das
859 und auch, dass es einige Abbruchbedingungen gibt. Natürlich weiß ich auf einen Blick jetzt nicht
860 genau, was die sind, ja. Anhand der Labels kann ich mir das aber dann schon anschauen. Also das
861 finde ich von der Übersichtlichkeit her, finde ich es schon ganz gut. Die Einfachheit ist dann aber,
862 glaube ich, doch ein bisschen anders. (...) Weil es halt/ Weil ich finde bei der Einfachheit muss
863 man dann schon das heranziehen, dass die Notation halt dazu führt, dass mal dort doch fünf
864 eigene Measure-Punkte modellieren muss. Und da ist die Frage, ob man das nicht, ich wüsste
865 jetzt auch nicht großartig wie, ja, nicht noch ein bisschen einfacher gestalten KÖNNTE.
866 #00:28:02-6#

867 175. I: Ja. #00:28:03-3#

868 176. B: Deswegen würde ich bei der Einfachheit auf weder noch/ Also ich finde es jetzt nicht gut,
869 nicht schlecht. Ich glaube mit Core-BPMN wäre es noch viel umständlicher, ja. Also da wäre die
870 Einfachheit überhaupt nicht gegeben, ja. Und ich weiß aber nicht, ob es nicht irgendwie noch ein
871 bisschen besser ginge, dass man dort noch ein bisschen das Ganze einfacher macht.
872 #00:28:35-4#

873 177. I: Hm, hm. (zustimmend) #00:28:37-0#

874 178. B: Dann von der Logik/ ja. Also. Da das System mit parallel und sequentiell ja immer das gleiche
875 ist, ja, ist das eigentlich klar, jetzt mittlerweile. Also da glaube ich kommst du einfach hinein. Da
876 macht auch die Komplexität von dem Modell keinen Unterschied, ja. //I: Ja. // Dass du dann
877 weißt, was läuft parallel, was läuft sequentiell, ja. Also, da denke ich, da gibt es dann nichts mehr.
878 Also, wenn du da ein paar Modelle gesehen hast und vielleicht mit jemandem durchgesprochen
879 hast, dann ist dir klar, das läuft parallel, das läuft sequentiell. Fertig, ja. Also das würde ich
880 mittlerweile, auch wenn ich glaube ich, vorhin habe ich gesagt, 'Gut ist es.', würde ich
881 mittlerweile auf sehr gut hoch gehen bei dem Modell. Weil es einfach/ mit der Zeit kommt man
882 da hinein und das glaube ich ist gar kein Thema mehr. Also. //I: Okay. // Das finde ich schon sehr
883 passend. #00:29:46-0#

884 179. I: Wunderbar, danke. Und letzter Punkt wäre Erweiterbarkeit. #00:29:51-0#

885 180. B: Puh, also. Da, Erweiterbarkeit, ja auch fast teils zusätzliche Informationen hineinzupacken, ja,
886 würde das wiederum gegen Verständlichkeit, Übersichtlichkeit und Einfachheit vielleicht sogar
887 sprechen, ja, würde ich eher sagen, dass ich da eher nichts mit hineinnehmen würde, noch
888 zusätzlich, ja. Also. Mir fällt jetzt auch nicht großartig was ein, was man dann noch hinzufügen
889 kann, ja. Und, da muss ich mit dem im Klaren sein, wenn man dann noch etwas zusätzlich/
890 Natürlich, man möchte immer so viele Informationen, wie möglich, ja, aber irgendwann geht es
891 halt einfach nicht mehr und es ist zu viel, ja. //I: Ja. // Und da finde ich das halt momentan sehr
892 ausgewogen, vielleicht eh schon ein bisschen viel Information. Aber noch ganz gut händelbar
893 (handhabbar), aber mehr Information/ Wenn man mehr Information möchte, dann kann man ja
894 das dann mit Interaktion lösen, indem man irgendwo drauf klickt und dann diese Variable zieht
895 und so weiter, ja. Aber in das Modell DA jetzt selbst in diese Darstellung würde ich da jetzt nicht
896 noch mehr mit hineinnehmen. #00:31:16-8#

897 181. I: Okay. #00:31:18-4#

898 182. B: Also, ich finde, das ist ein sehr gutes/ Also, da würde ich Sehr Gut nehmen. Würde sagen, es
899 ist ein sehr gutes Mittelmaß, wie das aussieht. Also, vielleicht sogar eher weniger Information,
900 wie noch mehr. Damit eben gerade die oberen Punkte mit Verständlichkeit, Übersichtlichkeit,
901 Einfachheit davon profitieren können. #00:31:47-0#

902 183. I: Okay, wunderbar, Dankeschön. Nach der Einführung, wie Sie gesagt haben, und nachdem Sie
903 diese Beispiele hier gesehen haben. Würden Sie sagen, dass Sie aufgrund dieser Erweiterungen
904 bereit sind, diese Modellierungsmethode in Ihrem Arbeitsalltag einzuführen, wenn Sie
905 kontinuierliche Prozesse modellieren müssten? #00:32:15-2#

906 184. B: Also, ich weiß jetzt nicht, ob wir demnächst irgendwann mit kontinuierlichen Prozessen zu tun
907 haben, aber an und für sich, da wir auch diese Engine verwenden, kann ich mir das schon gut
908 vorstellen, weil mir eben jetzt schon ganz klar ist, wie das funktioniert, ja. Also das ist der große
909 Vorteil, ja. Ich kann das ja dann einfach verwenden, ja, und ich kriege diese Measures und kann
910 dann Controls und Abbruchbedingungen definieren. Also ich denke von dem, würde ich das auf
911 jeden Fall verwenden. Also, von der Logik her ist es einfach, ja, ist es einfach. Wenn ich das jetzt
912 im Unternehmen auch wem erklären müsste, dann traue ich mir das nach dieser kurzen
913 Einführung hier zu, dass ich das jemandem erklären kann. Und ich traue der Person auch zu, dass
914 sie das dann auch versteht und auch anwenden kann, ja. Also das, denke ich, sollte kein Thema
915 sein. Von daher also würde ich schon das bevorzugen bevor ich da irgendwelche Workarounds
916 mit normalem BPMN machen muss, ja. Weil da muss ich wirklich, denke ich, in sehr viele Loops
917 gehen und dann hast du das Problem, dass du die/ da irgendwie heraus springen musst, meiner
918 Meinung nach, ja, wenn ich so einen Prozess betrachte. Und das macht das/ wenn du irgendwo
919 springen musst bei einem Modell, macht es das halt nicht einfacher, ja. //I: Ja. // Und deswegen
920 würde ich da schon dieses Modell definitiv bevor/ also diese Notation definitiv bevorzugen, ja.
921 #00:34:05-1#

922 185. I: Okay. Dankeschön. Jetzt kommen ein paar Fragen, die nächsten drei könnte man sagen, die
923 auf die Prozesse dahinter etwas abzielen. Und zwar. Wie gut würden Sie sagen, beschreiben die
924 Erweiterungen ein Kontrollsystem für diese Beispiele? Also Sie wissen, was grundsätzlich
925 abgebildet wird. Würden Sie sagen, dass diese Modelle, die Sie hier gesehen haben, mit den
926 Erweiterungen, diese Kontrollsysteme auch entsprechend abbilden können? Oder würden Sie
927 sagen, dass Ihnen jetzt auffällt, dass vielleicht etwas fehlt und dass man es vielleicht noch
928 detaillierter beschreiben könnte? Und die letzte Frage dazu wäre eigentlich schon Nummer
929 Neun. Wenn Sie Erfahrung in der Regelungstechnik haben, das weiß ich jetzt aktuell nicht, ob Sie
930 Erfahrung haben, würden Sie/ was würden Sie empfehlen, um diese Erweiterungen zu ergänzen,
931 um sie für Ingenieure, Regelungstechniker, Verfahrenstechniker attraktiver zu machen? Ich
932 nehme an, dass man Sie //B: Okay, also auf/ // Ich nehme an, man kann die Antworten gut hier
933 überleitend machen. Deswegen habe ich die drei jetzt gleich auf einmal gestellt. Also fangen wir
934 vielleicht mal grundsätzlich an, ob Sie meinen, dass die Prozesse dahinter auch gut abgebildet
935 wurden. In Modell Eins und Modell Zwei. #00:35:36-9#

936 186. B: Genau, also ich habe jetzt nochmal da das erste Modell offen. Und, ja, also ich denke, dass das
937 mit dieser Notation, dass/ Ich finde, das ist gut abgebildet. Dass dieses Modell genau das gleiche
938 sagt, wie im Prinzip das Flowchart oben, ja. Also, das finde ich schon, dass das Modell die quasi
939 reale Welt dann widerspiegelt, kann ich mir schon sehr gut vorstellen, ja. Also, da bleibt für mich
940 eigentlich dann nicht mehr viel, was mir unklar sein könnte, ja. Also mir ist es eigentlich klar,
941 wenn ich mir das Modell anschau, wie das ablaufen soll. Zumindest natürlich brauche ich schon
942 ein bisschen Einführung, worum es überhaupt geht und was für ein Thema, ja, //I: Ja. // aber
943 dann ist mir schon klar, wie das ablaufen soll, ja. #00:36:42-1#

944 187. I: Würden Sie das gleiche auch für Modell Zwei sagen, oder? #00:36:50-4#

945 188. B: Ja, also ich finde auch, dass das auch für das zweite Beispiel ganz gut passt. Also natürlich es
946 ist komplexer, ja, aber wenn das Beispiel komplexer ist, muss natürlich auch das Modell
947 komplexer werden. Ja. #00:37:11-6#

948 189. I: Und würden //B: (Hustet) Entschuldigung. // Sie sagen/ Nein, kein Problem, kein Problem.
949 Würden Sie sagen, dass etwas für eine detailliertere Prozessbeschreibung fehlt? Also aufgrund
950 der Prozesse, die Sie jetzt hier gesehen haben, also Temperaturregelung in beiden Fällen?
951 #00:37:32-5#

952 190. B: Da ich jetzt ja den Prozess auch nur von dieser kurzen Beschreibung eigentlich kenne, kann ich
953 da eigentlich nicht viel sagen. Also mir fehlt an und für sich nichts. Das Wichtigste, wie man die
954 Daten bekommt und welche Controls ausgeführt werden, das ist da, ja. Und ich finde, das ist die
955 Essenz, ja. Also das muss da sein. Und wenn das da ist, dann glaube ich, gibt es da sonst nichts
956 Anderes zu tun, ja. Was mich nur bisschen wundert ist, gibt es immer nur/ Also das ist jetzt gar
957 nicht auf diese Modelle bezogen, aber gibt es immer nur ein Control Event oder könnten das
958 auch mehrere sein? #00:38:24-8#

959 191. I: Das könnten auch mehrere sein. #00:38:26-6#

960 192. B: Könnten eh mehr? //I: Ja. // Okay. Dann habe ich das eh richtig verstanden, gut. #00:38:30-7#

961 193. I: Nur auf/ natürlich aufgrund der Leitfadenkonstruktion und damit die Komplexität halt nicht zu
962 hoch wird, jetzt im Zuge der Interviews, habe ich jetzt mich nur für diese beiden Beispiele
963 entschieden. Weil hier auch Unterlagen natürlich zur Verfügung gestanden sind, auf deren Basis
964 ich die Prozesse //B: Ja. // beschreiben kann. Deswegen auch, ja. #00:38:56-3#

965 194. B: Okay, ja, also ich würde sagen eben auch Modell Zwei ist auch sehr gut, sehr gut abgebildet, ja.
966 #00:39:03-5#

967 195. I: Frage Neun. Also auf Frage Neun beziehend. Würde Ihnen vielleicht etwas einfallen, um es für
968 Ingenieure, also für Steuerungstechniker oder Verfahrenstechniker attraktiver zu machen?
969 #00:39:18-2#

970 196. B: Da ich in dem Bereich keine Erfahrungen habe, wüsste ich jetzt nicht, was man für diese
971 Leute/ Ich meine, die kennen halt GANZ andere Notationen, ja, und deswegen weiß ich nicht.
972 Vielleicht hilft es, wenn man ihnen eine eigene Notation gibt. Aber das kann ich jetzt, das kann
973 ich so nicht beantworten. #00:39:45-2#

974 197. I: Okay, gut. Dankeschön. Wir kommen zur letzten größeren Frage. Ich werde Ihnen hier ein paar
975 kleinere Fragen stellen. Und da gibt es wieder die Bewertung von Eins bis Fünf, also Eins, sehr
976 schlecht, Fünf, sehr gut. Und zwar geht es jetzt im Allgemeinen um die Modellierung an sich.
977 Quasi wie einfach gewisse Dinge für Modellierer, für User, erkennbar oder durchführbar sind.
978 Die erste Frage wäre, wie einfach ist in den gezeigten Modellen nachzuvollziehen, dass die
979 einzelnen Abläufe parallel und unabhängig voneinander laufen. #00:40:29-8#

980 198. B: Ja, also, ich glaube, das habe ich schon mittlerweile auch mehrfach betont. Also ich finde das
981 mittlerweile sehr gut nachzuvollziehen. Es ist mir eigentlich klar, was läuft parallel ab und was
982 läuft sequentiell ab, also. #00:40:45-5#

983 199. I: Das heißt, Sie würden Gut oder Sehr gut vergeben? #00:40:53-1#

984 200. B: Genau. Ich würde Sehr gut vergeben. Also mittlerweile. Am Anfang, wäre vielleicht/ Ganz am
985 Anfang wäre ich zwischen Weder noch und Gut geschwankt, dann war ich auf Gut, aber
986 mittlerweile bin ich da auf Sehr gut. #00:41:04-6#

987 201. I: Okay, danke. Zweite Frage. Wie einfach ist es zu definieren, wann eine Anpassung oder
988 Regulierung am System erfolgt? #00:41:15-4#

989 202. B: Inwiefern ist da das Wort Definieren/ also für mich als Modellierer, wie einfach dass ich das
990 machen kann? Oder? #00:41:34-7#

991 203. I: Genau. Wie einfach können Sie zeitliche Bedingungen quasi oder generelle Bedingungen für
992 Regelungen definieren in so einem Prozessmodell? #00:41:45-6#

993 204. B: Ja, ich glaube, das ist super einfach, weil ich muss ähm nur quasi diese Aktivität, dieses Symbol,
994 hinzufügen und danach meine Tasks anfügen. Also ich glaube, das sollte von der Komplexität her
995 wenn man Modellierungserfahrung hat, kein Thema sein. Also ich glaube, das ist auch sehr gut
996 durchzuführen, weil ich sehe da jetzt kein Hindernis. #00:42:17-0#

997 205. I: Okay. Dritte Frage. Wie einfach ist es, die maximale Dauer einer Anpassung zu definieren?
998 #00:42:27-7#

999 206. B: Ich habe das jetzt ja nicht gemacht, aber ich glaube, das ist ja nur entweder einen Wert setzen
1000 oder ein Dropdown-Menü oder? Also. #00:42:39-8#

1001 207. I: Genau, in dem Fall ist es/ // B: Wo man dann/ // Ja. #00:42:42-1#

1002 208. B: Ich glaube, momentan würde man halt einfach den Wert dafür setzen. Das heißt, ich stelle mir
1003 das so vor. Man klickt drauf auf diese Zustandsabfrage, und dann setzt man dort einfach den
1004 Wert, oder? #00:42:58-8#

1005 209. I: Genau, das wäre das Vorgehen, ja. #00:43:00-9#

1006 210. B: Ja. (lacht) Dann denke ich, dass das doch sehr einfach abläuft, also. #00:43:08-4#

1007 211. I: Nächste Frage. Wie einfach ist es zu definieren, unter welchen Bedingungen sämtliche
1008 repetitiven Aufgaben beendet werden sollen? #00:43:20-1#

1009 212. B: Also ich finde die Fragen schwieriger wie die Antworten. // I: (lacht) // Also wie einfach ist es
1010 zu definieren, unter welchen Bedingungen/ #00:43:30-5#

1011 213. I: Da geht es darum, wenn Sie die Aufgabe hätten zu definieren, welche Bedingungen/ Also jetzt
1012 bei dieser konkreten Frage, wenn Sie jetzt sich denken, ich muss da jetzt hinein modellieren //B:
1013 Ach so. // in das Prozessmodell/ #00:43:43-4#

1014 214. B: Geht es darum/ Dann geht es darum/ Da geht es um dieses Wait und Cancel, nehme ich an?
1015 #00:43:46-7#

1016 215. I: Ja, auch. DAS in Zusammenhang mit den Abbruchbedingungen. #00:43:53-8#

1017 216. B: Ach so, um die Abbruch/ Okay. #00:43:57-2#

1018 217. I: Also wenn Sie sich denken, 'Ich habe jetzt die Aufgabe, Bedingungen zu setzen, dass sämtliche
1019 repetitiven Aufgaben, also die Dinge, die sich immer wiederholen, das ich //B: Ja, ja, ja. // die
1020 beende.' #00:44:09-3#

1021 218. B: Jetzt habe ich es verstanden. Okay. #00:44:10-8#

1022 219. I: Wie mache ich das? Also so ist das eher abgezielt. Ich habe eine gewisse Aufgabe im Kopf, die
1023 ich modellieren möchte, oder einen gewissen Verlauf, den ich modellieren möchte. Und wie
1024 einfach wäre es für mich, wenn ich weiß, wie diese Erweiterungen funktionieren? #00:44:26-3#

1025 220. B: Ja, also ich denke, das ist genau das gleiche wie vorher, ja. Also ich würde/ Also diese
1026 Abbruchbedingungen jetzt einzufügen ist/ Von der Modellierungslogik, denke ich, funktioniert
1027 das einfach, ja. Die Komplexität könnte natürlich dann durch die Bedingung selbst kommen, aber
1028 das tut hier nichts zur Notation bei, ja. Also ich denke, von der Notation her funktioniert das sehr
1029 gut, also sehr einfach. //I: Okay. // Und wie komplex ich das dann selber gestalten möchte, liegt
1030 ja dann am Modellierer, ja, also wie die Abbruchbedingung dann stattfinden soll. #00:45:18-9#

1031 221. I: Ja, ja. Vorletzte Frage. Wie einfach ist es für den Modellierer zu definieren, dass nachdem
1032 repetitive Aufgaben beendet werden sollen, also Abbruchbedingungen quasi erfüllt sind, danach
1033 Aufräumarbeiten einmalig zu definieren? Dass die einmalig zu erfolgen haben? #00:45:43-5#

1034 222. B: Also grundsätzlich einmal finde ich, dass es sehr leicht ist. Ich finde es auch spannend, dass die
1035 Aufräumarbeiten bei den Modellen direkt immer nach den Abbruchbedingungen sind. Weil ich
1036 könnte mir auch ein gutes Beispiel vorstellen, wo einfach die Aufräumarbeiten nach der Closed
1037 Loop stattfinden. #00:46:05-2#

1038 223. I: Ja. #00:46:06-6#

1039 224. B: Also, nur so. Also ich finde, das sind zwei Möglichkeiten, die man ins Auge fassen kann, aber
1040 sind natürlich leicht durchzuführen, also beides. Ich sehe da keine Hindernisse, ja. Das ist eine
1041 Aktivität hinzufügen und fertig ist das, ja. Also. #00:46:23-6#

1042 225. I: Genau. Wie Sie richtig sagen. Also es soll dem User freigestellt sein, auf spezifische
1043 Bedingungen spezifische Aufräumroutinen zu definieren oder quasi allgemeingültige nach der
1044 Closed Loop. #00:46:35-8#

1045 226. B: Genau, genau. //I: Ja. // Also man wird beides mal verwenden können, also. Zum Beispiel,
1046 wenn man sagt, 'Okay, da oben jetzt. Man switched zum Schluss immer das System auf off', ja.
1047 Dann könnte man das natürlich immer nach der Closed Loop machen. Was natürlich nicht geht
1048 in den oberen Beispiel, ja, weil man hat den Operation Mode. Wenn der auf manual geschaltet
1049 wird, möchte man natürlich nicht, dass bei den Aufräumarbeiten dann zum Schluss das System
1050 heruntergefahren wird. Also. Eh klar. #00:47:14-5#

1051 227. I: Und die letzte Frage ist, wie einfach ist es, komplexe Abläufe im Kontext von kontinuierlichen
1052 Prozessen mit diesen Erweiterungen zu beschreiben. Das soll ein bisschen darauf abzielen, dass
1053 wenn man den doch vielleicht etwas schwieriger zu verstehenden Charakter hinter
1054 kontinuierlichen Prozessen, wenn man jetzt ein komplexeres Modell hätte, wie das Zweite, dass
1055 Sie gesehen haben, dass man so etwas generell modellieren kann mit den Erweiterungen. Also
1056 für die Verständlichkeit. Welche Messungen brauche ich und wie reagieren die entsprechenden
1057 Reglermodelle darauf und welche Abbruchbedingungen gibt es? Also das Ganze allgemein
1058 gesehen, könnte man hier sagen. #00:48:02-9#

1059 228. B: Ja, also. Ich finde das allgemein/ finde ich es sehr gut, ja. Also es ist mir einfach, und das muss
1060 man schon sagen, ja. BPMN ist jetzt kein kleines Werkzeug, ja, sondern, da gibt es schon ein
1061 Riesenrepertoire, ja. Und das jetzt mit im Prinzip eins, zwei, drei, vier Symbolen, ja, so zu
1062 erweitern, dass ich das machen kann, das ist schon sehr, sehr stark, ja. Also weil das bietet
1063 unheimlich viele Möglichkeiten, ja. Vielleicht ja nicht nur für kontinuierliche Prozesse, ja. Da

1064 muss man dann halt schauen, ja. Aber ich denke, das ist schon eine wirklich mächtige
1065 Erweiterung, ja. Und sie erleichtert das halt so ungemein, weil wenn man jetzt das mit normalen
1066 BPMN abbilden möchte, denke ich, wird man wirklich daran verzweifeln, ja. Das wird einfach von
1067 der Komplexität her wahrscheinlich ein Wahnsinn werden. Vor allem, weil man eben auch
1068 ungelöste Probleme hätte, wie eben das, dass man dann aus Loops herausspringen muss zu
1069 anderen Aktivitäten. Also quasi aus der Programmierung kennt man das mit Go to. Und ich
1070 glaube, das wäre wirklich, wirklich schwer, ja. Also von daher finde ich die Lösung auch beim
1071 komplexen Beispiel als sehr gut, ja, vor allem wenn ich es mit BPMN vergleichen würde.
1072 #00:49:46-3#

1073 229. I: Okay. Dankeschön. Ja, das war es schon vom Leitfaden her, für das Interview. Danke nochmal,
1074 dass Sie sich Zeit genommen haben, ich bin auch/ #00:50:01-7#

1075 230. B: Ja, bitte. Gerne. #00:50:01-9#

1076 231. I: Vielen Dank. Ich bin auch sehr dankbar, wenn Sie mir Feedback geben möchten, zum Leitfaden.
1077 Also, wenn Sie das Gefühl haben, dass eventuell manche Fragen zu komplex gestellt sind oder zu
1078 lange dauern oder generell das Interview eventuell schon unangenehm lange gedauert hat.
1079 Dann können Sie mir das gerne sagen. Ich bin immer dankbar dafür, wenn ich auch Feedback
1080 kriege. Und die Aufnahme würde ich dann auch gleich beenden. Ab jetzt.
