Hochschule
Bonn-Rhein-Sieg
University
of Applied Sciences

Fachbereich Informatik
Department of Computer Sciences

# Masterarbeit

Studiengang Informatik (M.Sc.)

# Vergleich von BPMN basierten WfMS mittels Resource Patterns

von

Daniel Reiß

| | |
|---|---|
| Matrikelnummer | 9033783 |
| E-Mail | daniel.reiss@smail.inf.h-brs.de |

| | |
|---|---|
| Erstprüfer | Prof. Dr. Hense |
| Zweitprüfer | Prof. Dr. Bertram |

| | |
|---|---|
| eingereicht am | 04.10.2024 |

## Eidesstattliche Erklärung

Name: Daniel Reiß

Adresse: Johannesstraße 67, 53225 Bonn

**Erklärung**

Hiermit erkläre ich an Eides Statt, dass ich die vorliegende Arbeit selbst angefertigt habe; die aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht.

Die Arbeit wurde bisher keiner Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.

Unterschrift Sankt Augustin, den

## Inhaltsverzeichnis

## Abbildungsverzeichnis

## Abkürzungsverzeichnis

| | |
|---|---|
| Business Process Model and Notation | BPMN |
| Workflow Management System | WfMS |
| Business Process Management System | BPMS |
| Business Process Execution Language | BPEL |
| Graphical User Interface | GUI |
| Decision Model and Notation | DMN |
| Unified Modeling Language | UML |
| Case Management Model and Notation | CMMN |
| Application Programming Interface | API |
| Create, Read, Update, Delete | CRUD |
| Lightweight Directory Access Protocol | LDAP |
| Representational State Transfer | REST |
| Enterprise Resource Planning | ERP |
| Robotic Process Automation | RPA |
| Machine Learning | ML |
| Artificial Intelligence | AI |

# 1 Einleitung

## 1.1 Motivation und Problemstellung

Das Ziel der vorliegenden Arbeit „Vergleich von BPMN basierten WfMS mittels Resource Patterns“ ist es, zu untersuchen, wie sich die Resource Patterns in ausgewählten Business Process Model and Notation (BPMN) basierten Workflow Management Systemen (WfMS) umsetzen lassen, um anhand dessen die untersuchten Tools zu vergleichen. Hierfür wird die zentrale Fragestellung „Wie sind Resource Patterns in den heutigen BPMN basierten WfMS umgesetzt?“ untersucht. Um diese zentrale Fragestellung zu beantworten, werden für jedes Resource Pattern aus [93], die folgenden Teilfragen betrachtet:

- Q1: Kann das Resource Pattern in den heutigen BPMN basierten WfMS abgebildet werden?
- Q2: Wenn ja, wie gut lässt sich das Resource Pattern in den heutigen BPMN basierten WfMS abbilden?

Die zentrale Fragestellung ist relevant, da ein WfMS in der Lage sein muss, die Resource Patterns umzusetzen, um einen Workflow umfassend abbilden und ausführen zu können [93, 110]. Ein Workflow ist komplex und um ihn umfassend zu betrachten, müssen viele verschiedene Aspekte berücksichtigt werden [93]. Diese Aspekte lassen sich in drei übergreifende Bereiche, auch Perspektiven genannt, gruppieren: die Control-Flow Perspektive, die Data Perspektive und die Resource Perspektive [93]. In jeder dieser Perspektiven gibt es sogenannte Workflow Patterns, welche die wichtigsten Konzepte und wiederkehrende Anforderungen eines Workflows in der jeweiligen Perspektive darstellen [93, 110]. Ein WfMS, in dem solche Workflows bzw. Geschäftsprozesse definiert, ausgeführt und verwaltet werden können, muss dementsprechend in der Lage sein, die Patterns der verschiedenen Perspektiven umzusetzen, damit ein Workflow umfassend abgebildet und ausgeführt werden kann [93, 110]. Hierfür benötigt ein WfMS ein Modell des Workflows, welches mithilfe einer Modellierungssprache angefertigt werden muss [93].

In der Literatur wird allerdings darauf verwiesen, dass WfMS ihren Fokus eher auf die anderen beiden Perspektiven legen, wodurch die Resource Perspektive vernachlässigt wird [93, 101]. Dies wird durch die Ergebnisse aus [84, 93, 110] bestätigt, bei denen die betrachteten WfMS einen Großteil der Resource Patterns nicht umsetzen konnten. Seitdem wurden keine Untersuchungen durchgeführt, welche WfMS mithilfe der Resource Patterns analysieren und somit die Ergebnisse aus [84, 93, 110] erneuern. Daher verweisen auch aktuelle Quellen weiterhin auf die mittlerweile über 15 Jahre alten Ergebnisse von [84, 110]. Wie in dem Kapitel „2.4.2 Resource Patterns und deren Kontext in Modellierungssprachen und WfMS“ dargestellt, ist die Vernachlässigung der Resource Perspektive bzw. Resource Patterns in Modellierungssprachen, welche WfMS nutzen müssen, allerdings immer noch aktuell. Demnach ist es relevant die Ergebnisse aus [84, 93, 110] zu erneuern und dafür die Resource Perspektive in heutigen BPMN basierten WfMS zu betrachten, indem die Umsetzung der Resource Patterns analysiert wird. Die Resource Patterns dienen hier als etabliertes Analysewerkzeug, um die Resource Perspektive sowohl in WfMS als auch in Modellierungssprachen zu bewerten [22]. Der Fokus auf BPMN wird bewusst gewählt, da sich die Modellierungssprache als Industriestandard etabliert hat [108, 109], aber in den neusten Untersuchungen weiterhin Defizite bezüglich der Resource Perspektive aufweist [93].

## 1.2 Aufbau und Vorgehen

Um die zentrale Fragestellung „Wie sind Resource Patterns in den heutigen BPMN basierten WfMS umgesetzt?“ in der vorliegenden Arbeit zu beantworten, werden für jedes Resource Pattern die Teilfragen Q1 und Q2 betrachtet. Als erstes werden hierfür in dem Kapitel „2 Grundlagen der Arbeit“ die zentralen Begriffe und Konzepte aus den

Themenfeldern WfMS, BPMN und Resource Patterns erläutert. Die einzelnen Resource Patterns werden dabei in dem Kapitel „2.3 Definition, Kontext und Vorstellung der Resource Patterns“ erläutert. Dabei sind die Beispiele für jedes Pattern Ausschnitte aus den Beispielworkflows, welche im Rahmen der Untersuchung in den Tools umgesetzt werden. Diese Ausschnitte bilden die Funktionsweise des Patterns ausreichend ab. Anschließend wird in den Unterkapiteln von „2.4 Stand der Forschung und resultierende Relevanz der Arbeit“ die Literaturrecherche beschrieben und mit den identifizierten Quellen die Relevanz des Themas und der Arbeit belegt. Daraufhin wird in dem Kapitel „2.5 Auswahl und Überblick der untersuchten Tools“ die Vorgehensweise erläutert, mit der die zu analysierenden Tools ausgewählt werden, um in den anschließenden Unterkapiteln die konkreten Tools kurz vorzustellen. Dabei handelt es sich um die folgenden ausgewählten Tools:

- Axon Ivy (Axon Ivy Designer 11.2.1)
- Bonitasoft (Bonita Studio Community Edition 2023.2)
- Camunda (Camunda 7 Community (Version 7.21))

Darüber hinaus wird auch YAWL (YAWL 5.0) erläutert und untersucht, allerdings außer Konkurrenz, da es kein BPMN verwendet. Demnach werden die Endergebnisse für YAWL nur in den Abbildungen 7 bis 14 angegeben und in dem Kapitel „3.3.1 Vergleich der Tools aufgrund der Untersuchungsergebnisse der Patterns“ kurz beschrieben. Da YAWL auf Grundlage der Workflow Patterns erstellt wurde [93, 105], wird es als Beispiel betrachtet, um zu sehen, wie die Resource Patterns konkret umgesetzt werden könnten und ob selbst YAWL manche Resource Patterns nicht abbilden kann. Eine genaue Erläuterung zu der Rolle und Verwendung von YAWL in der vorliegenden Arbeit und eine kurze Vorstellung des Tools werden in dem Kapitel „2.5.4 Vorstellung von YAWL als zusätzliches Beispieltool“ hinterlegt. In dem Kapitel „3 Untersuchung der Resource Patterns in den ausgewählten Tools“ erfolgt die konkrete Untersuchung der Tools, wodurch die zentrale Fragestellung samt ihrer Teilfragen Q1 und Q2 beantwortet werden soll. Zu Beginn wird das konkrete Bewertungsschema vorgestellt, welches verwendet wird, um jedem Tool pro Pattern eine Gesamtbewertung von 0 bis 10 Punkten zu vergeben, je nachdem wie gut sie das Pattern abbilden können. Eine genaue Beschreibung erfolgt in dem Kapitel „3.1 Erläuterung des Bewertungsschemas für die Untersuchung“. Daraufhin wird die Untersuchung der Resource Patterns in den drei Tools in dem Kapitel „3.2 Durchführung der Untersuchung samt Bewertung der Patterns“ konkret beschrieben. Aus Gründen der Übersichtlichkeit wurde die Untersuchung in Unterkapitel aufgeteilt, die jeweils Patterns aus einer Kategorie betrachten. Für jedes Pattern werden zuerst die zu untersuchenden Kriterien aufgelistet, um anschließend zu beschreiben und mit dem Bewertungsschema zu evaluieren, wie gut sich diese in dem jeweiligen Tool umsetzen lassen. Für die Umsetzung wird pro Pattern ein Beispielworkflow gewählt, welcher in allen drei Tools implementiert wird. Diese Beispielworkflows werden aus den jeweiligen Flash-Animationen der Resource Patterns von der offiziellen Workflow Pattern Webseite übernommen [114]. Die Unterseiten für jedes einzelne Resource Pattern, welche in [114] verlinkt werden, enthalten die jeweiligen Flash-Animationen. Dadurch wird sichergestellt, dass die Resource Patterns in allen Tools einheitlich dargestellt und untersucht werden. Eine detaillierte Beschreibung der Beispielworkflows wird nicht vorgenommen, da diese nur als Mittel genutzt werden, um die Kriterien einheitlich überprüfen zu können. Der Inhalt der Workflows ist dabei nicht relevant. Bei Bedarf sind Abbildungen der Beispielworkflows und die einzelnen Implementierungen im digitalen Angang zu finden. Die vollständigen Animationen der Beispielworkflows können, wie oben beschrieben, auf den Unterseiten von [114] für die einzelnen Resource Patterns betrachtet werden. Abschließend wird in dem Kapitel „3.3.1 Vergleich der Tools aufgrund der Untersuchungsergebnisse der Patterns“ zusammengefasst wie viele Punkte jedes Tools durchschnittlich für die Resource Patterns erhalten hat, um die Tools zu vergleichen. Daraufhin werden die Stärken und Schwächen der jeweiligen Tools, die in der Untersuchung aufgefallen sind, dargestellt. Anschließend werden in dem Kapitel „3.3.2 Bewertung der Antworten auf Forenbeiträge“ die Antworten auf Forenbeiträge bewertet, die während der

Untersuchung erstellt wurden. Es wird betrachtet, ob die Antworten neue Aspekte liefern und wie deren Qualität und Häufigkeit ist. Als letztes werden in dem Kapitel „3.3.3 Erkenntnisse über die Tools ohne direkten Bezug auf die Patterns“ Aspekte der Tools beschrieben, welche sich auf kein Pattern beziehen, aber trotzdem erwähnenswert für den Vergleich bzw. für eine Toolauswahl sind. In dem letzten Kapitel „4 Fazit und Ausblick“ werden die Ergebnisse der vorliegenden Arbeit zusammengefasst, um die zentrale Fragestellung samt ihrer Teilfragen zu beantworten. Außerdem wird in einem Ausblick geschildert, wie sich die Resource Patterns und WfMS in der Zukunft entwickeln könnten und welche Aspekte in zukünftigen Arbeiten weiter untersucht werden könnten. Sämtliche Abbildungen aus der vorliegenden Arbeit sind für eine bessere Lesbarkeit zusätzlich in voller Größe im digitalen Anhang auf dem USB-Stick vorhanden.

# 2 Grundlagen der Arbeit

## 2.1 Definitionen, Funktionsweise und Kontext eines WfMS

Ein Geschäftsprozess ist eine Reihe von miteinander verbundenen Aktivitäten, die gemeinsam ein Unternehmensziel verwirklichen, normalerweise im Rahmen einer Organisationsstruktur, die funktionale Rollen und Beziehungen definiert [112]. Dabei kann die Anzahl an verbundenen Aktivitäten beliebig sein, solange sie mindestens eins beträgt [112]. Ein Workflow wird definiert als die Automatisierung eines solchen Geschäftsprozesses, bei der Dokumente, Informationen oder Aufgaben nach bestimmten Regeln von einem Beteiligten an einen anderen weitergegeben werden [112]. Bei „Workflow“ und „Geschäftsprozess“ handelt es sich somit um zwei verschiedene Begriffe, aus Gründen der Verständlichkeit, werden sie in der vorliegenden Arbeit allerdings synonym verwendet. Dies orientiert sich auch am Vorgehen in der Praxis, da sowohl [84] als auch [93] die Begriffe synonym verwenden. [84] definiert ein Workflow Modell als die Beschreibung bzw. das Modell eines Geschäftsprozesses und setzt somit Workflow und Geschäftsprozess gleich und [93] führt keine Unterscheidung der beiden Begriffe durch. Zur Vereinfachung wird „Geschäftsprozess“ im Folgenden oft durch „Prozess“ abgekürzt. Ein System, in dem solche Workflows bzw. Geschäftsprozesse definiert, ausgeführt und verwaltet werden können, wird Workflow Management System (WfMS) genannt [112]. Hierfür setzt das WfMS Software ein, die in der Lage ist, die Prozessdefinitionen zu interpretieren, mit den Workflow-Teilnehmern zu interagieren und falls erforderlich IT-Werkzeuge und -Anwendungen zu nutzen [112]. Der Begriff WfMS stammt aus den 1990er-Jahren und mittlerweile werden WfMS auch als Business Process Management System (BPMS) bezeichnet [93, 101, 107]. Dabei haben BPMS typischerweise einen größeren Funktionsumfang und unterstützen neben den klassischen WfMS-Funktionen, wie die Prozessmodellierung und -ausführung auch Business Rules, Content Management oder auch Prozessanalyse-Funktionen wie Process Monitoring [93, 101, 102]. BPMS bieten somit umfassende Unterstützung für den Entwurf, die Bereitstellung, die Überwachung und die fortlaufende Verbesserung von Geschäftsprozessen [93, 102]. In der vorliegenden Arbeit sollen allerdings die Resource Patterns analysiert werden, welche sich auf die klassischen WfMS-Funktionen beziehen. Systeme die potenziell analysiert werden sollen, benötigen somit nur diese Funktionen bzw. auch nur diese Funktionen werden betrachtet. Demnach wurde sich für die vorliegende Arbeit entschieden, diese Systeme als WfMS zu betiteln. Hierunter fallen sämtliche Systeme, die über die klassischen WfMS-Funktionen verfügen, also einen Workflow mit allen drei Perspektiven zur Entwurfszeit modellieren und zur Laufzeit ausführen zu können. Der Fokus wurde hierbei auf die Resource Perspektive gelegt.

Der Workflow wird in einem solchen WfMS zur Entwurfszeit durch ein Workflow Modell abgebildet, welches die Funktionsweise des Prozesses definiert [84, 93, 101]. Die Modelle werden in einer bestimmten Modellierungssprache bzw. Notation, wie z.B. BPMN modelliert [93, 109, 111]. Eine detaillierte Erklärung für BPMN folgt in dem Kapitel „2.2 Definitionen, Einordnung und relevante Modellierungselemente von BPMN“. Die einzelnen Aktivitäten, aus denen der Prozess besteht, werden als Tasks bezeichnet [84, 93]. Diese sind durch gerichtete Pfeile, die die verschiedenen Ausführungspfade durch den Prozess angeben, verbunden [84, 93]. Eine Task repräsentiert dabei einen einzelnen Arbeitsschritt [84, 93]. Während der Ausführung eines solches Workflow Modells, also zur Laufzeit, wird für jede Task eine Instanz erzeugt, wenn diese Task abgerufen wird bzw. der Prozess an der Task angelangt ist [84, 93]. Diese Instanz wird Workitem genannt [84, 93]. Ein sog. Case ist ein Durchgang des Modells während der Ausführung, vom Startpunkt bis zum Endpunkt [84, 93]. Für jede Task die in einem Case durchlaufen werden muss, wird ein Workitem erzeugt [84, 93]. Spezielle Workitems müssen von menschlichen Ressourcen ausgeführt bzw. abgearbeitet werden [84, 93, 101]. Auf diese Workitems wird sich in der vorliegenden Arbeit fokussiert. Eine Ressource wird grundsätzlich als eine Einheit betrachtet, die in der Lage ist, Arbeit zu verrichten [93]. Somit können Ressourcen sowohl menschlich als auch nicht-menschlich sein [84, 93]. [93] und

[84] betrachtet nur menschliche Ressourcen auch in Bezug auf die Resource Patterns, weshalb auch diese Arbeit nur menschliche Ressourcen sog. User betrachtet.

Geschäftsprozesse werden grundsätzlich in dem Kontext einer Organisation bzw. eines Unternehmens definiert und die menschlichen Ressourcen der Organisation werden genutzt, um die Workitems abzuarbeiten [84, 93]. Daher ist ein sog. Organisationsmodell notwendig, um die Struktur der Organisation und für den Geschäftsprozess relevante Details über die Ressourcen und deren Beziehungen untereinander zu beschreiben [84, 93, 101]. Daher ist typischerweise jeder Geschäftsprozess mit einem solchen Organisationsmodell verknüpft [84, 93]. Klassisch können in einem Organisationsmodell die Rollen, die Fähigkeiten und die Gruppen einer Ressource festgelegt werden [84, 93, 101]. Eine Rolle beschreibt dabei Ressourcen mit ähnlichen Aufgaben oder Verantwortungsstufen und dient als Gruppierungsmechanismus [84, 93]. Jeder Ressource können Attribute beigefügt werden, die die Fähigkeiten der Ressource beschreiben [84, 93]. Eine Ressource kann über beliebig viele Rollen oder Fähigkeiten verfügen [84, 93]. Eine Gruppe beschreibt eine feste Gruppe an Ressourcen in einer Organisation, welche ähnliche Aufgaben erledigen, wie z.B. eine Abteilung [84, 93]. Gruppen können ineinander verschachtelt werden, um Hierarchien abzubilden, da eine Gruppe aus verschiedenen Untergruppen bestehen kann [84, 93]. Diese Konzepte können innerhalb einer Organisation verwendet werden [84, 93]. In einem Organisationsmodell können ggf. auch mehrere Organisationen modelliert werden [84, 93]. Die Informationen aus einem solchen Organisationsmodell können verwendet werden, um Workitems passenden Ressourcen zuzuordnen [84, 93, 101].

Die Ressourcen erhalten die Workitems durch den sog. Worklist handler [93, 101]. Der Worklist handler ist eine Softwareanwendung, welche die Ressource über die auszuführenden Workitems informiert und die Möglichkeit bietet, mit der Engine zu interagieren, um Workitems anzunehmen, zu verwalten und abzuarbeiten [93]. Die Engine ist dabei das System, das die gesamte Prozessausführung steuert und jeden Case, der initiiert wird, verwaltet [93]. Der Worklist handler zeigt die Workitem auf der sog. Worklist an [93]. Je nach WfMS kann eine Ressource eine oder mehrere Worklists haben oder auch auf globale Worklists zugreifen [93, 110]. Auch der Begriff Workqueue wird häufig für diese Liste an Workitems verwendet [84, 93]. Allerdings konnte keine klare Definition und/oder Trennung der beiden Begriffe in [84] oder [93] identifiziert werden. Darüber hinaus ist Workqueue, laut [112], ein Synonym für Worklist. Demnach werden die beiden Begriffe auch in der vorliegenden Arbeit synonym verwendet.

Häufig verfügt ein Workitem für die Interaktion mit den Ressourcen während der Ausführung über sog. Formulare [20, 32, 54, 93]. Diese Formulare dienen zur Erfassung von Daten oder zur Anzeige von Informationen, die für das Workitem relevant sind [20, 32, 54, 93].

Ein Workitem kann verschiedene Status haben [84, 93, 110]. Der Übergang von einem Status zum anderen wird Zustandstransfers genannt [84, 93, 110]. Grundsätzlich entsprechen viele Resource Patterns den Zustandstransfers [84, 93]. Für die Patterns, bei denen dies der Falls ist, wird der passende Zustandstransfer in dem Kapitel „3.2 Durchführung der Untersuchung samt Bewertung der Patterns“ jeweils benannt. Der Name eines jeden Zustandstransfers startet entweder mit dem Präfix „S:“ oder „R:“ [84, 93, 110]. Dies zeigt an, ob der Zustandstransfers von dem System (S:), also dem WfMS, oder von einer Ressource (R:), also dem User, initiiert wurde [84, 93, 110]. Die Zusammenstellung der möglichen Status und Zustandstransfers wird Workitem Lifecycle genannt [84, 93, 110]. Der grundlegende Workitem Lifecycle aus [93] wird in der Abbildung 1 dargestellt.

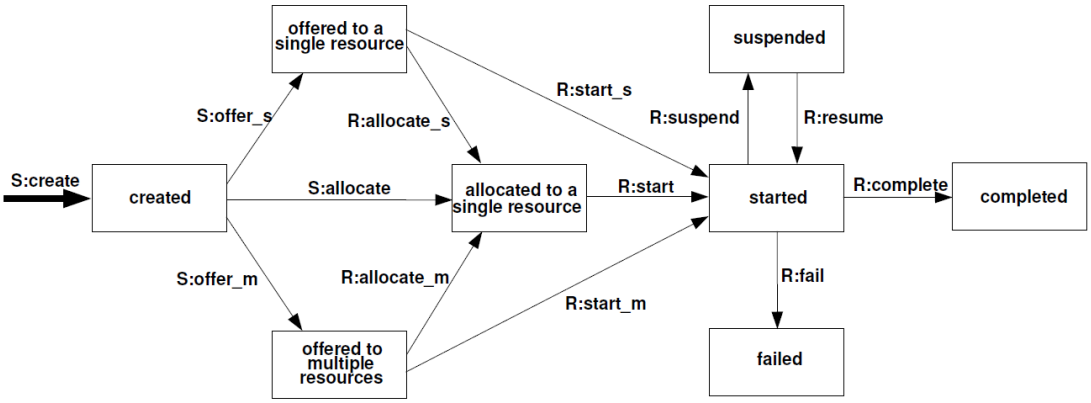


Abbildung 1: Workitem Lifecycle [93]

Der Workitem Lifecycle aus [93] wird auch in den Quellen [84, 110] benannt. Die Quelle [93] und damit auch die Abbildung 1 beschreiben eine allgemeine Definition des Workitem Lifecycles. Der Workitem Lifecycle kann je nach Tool und Umsetzung unterschiedlich sein. Die Abbildung 1 beschreibt den Workitem Lifecycle folgendermaßen [93]: Nachdem ein Workitem erstellt wurde, befindet es sich im Status „created“. Aus diesem Status kann das System das Workitem entweder einer Ressource oder mehreren Ressourcen anbieten. Dadurch wird es aus dem Status „created“ in den Status „offered“ versetzt. Eine Ressource muss ein Workitem, welches ihr angeboten wird, erst annehmen, bevor es ihr zugeteilt ist. Durch das Annehmen ändert sich der Status des Workitems von „offered“ auf „allocated“. Außerdem kann das System ein Workitem einer Ressource direkt zuteilen, wodurch sich der Status des Workitems von „created“ auf „allocated“ ändert. Ein Workitem kann immer nur einer Ressource gleichzeitig zugeteilt sein. Sowohl aus dem Status „offered“ als auch aus dem Status „allocated“ kann ein Workitem von einer Ressource gestartet werden. Nach dem Starten befindet sich das Workitem in dem Status „started“ und ist der Ressource zugeteilt, die es gestartet hat. Ein gestartetes Workitem kann pausiert werden, wodurch es in den „suspended“-Status wechselt. Nachdem es wieder fortgesetzt wird, wechselt der Status zu „started“ zurück. Sollte die Ressource das gestartete Workitem erfolgreich abschließen, wird es in den „completed“-Status versetzt. Falls das Workitem fehlschlägt, wird der Status von „started“ auf „failed“ geändert.

Weitere Zustandstransfers werden durch die Detour Patterns und Auto-Start Patterns in [93] ergänzt und sind in der Abbildung 2 und in der Abbildung 3 dargestellt.

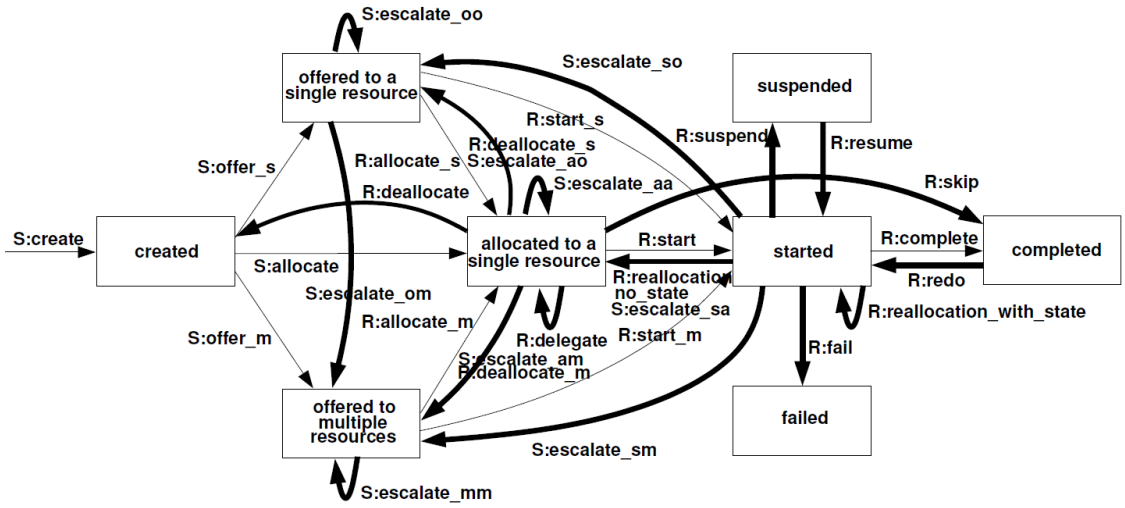


Abbildung 2: Ergänzung Workitem Lifecycle (Detour Patterns) [93]

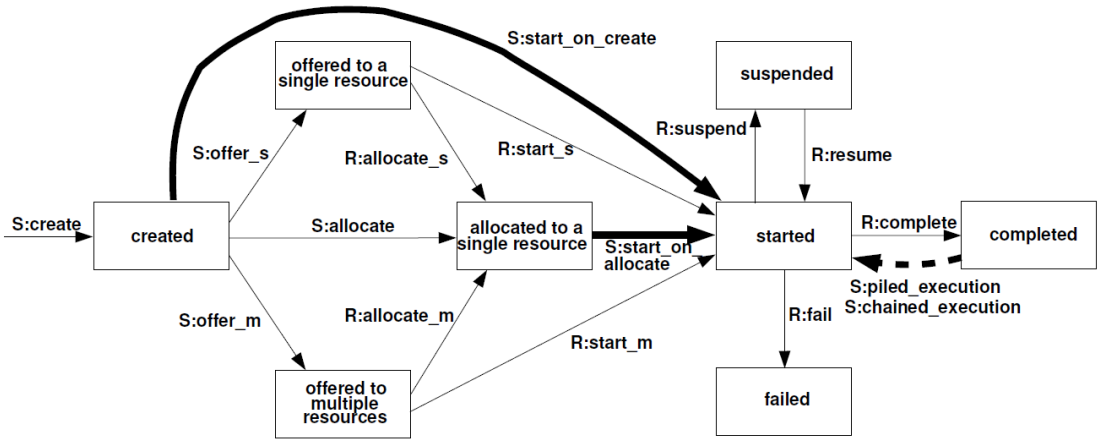


Abbildung 3: Ergänzung Workitem Lifecycle (Auto-Start Patterns) [93]

An diesen drei Abbildungen können sämtliche Status und Zustandstransfers abgelesen werden. Die zusätzlichen Zustandstransfers der Abbildung 2 und der Abbildung 3 entsprechen den jeweiligen Detour und Auto-Start Patterns und werden daher bereits in den Kapiteln „3.2.4 Untersuchung der Detour Patterns“ und „3.2.5 Untersuchung der Auto-Start Patterns“ genauer beschrieben, weswegen sie hier nicht erläutert werden.

Die deutschen Übersetzungen für viele der feststehenden Begriffe im WfMS-Kontext sind nicht trennscharf und könnten mehreren Begriffen zugeordnet werden. Daher wurden für die vorliegende Arbeit folgende Übersetzungen festgelegt:

- offer = anbieten, offerieren
- claim = annehmen
- allocate = zuteilen, allokieren
- assign = zuordnen
- execute = ausführen
- start = starten
- map = zuweisen
- deallocate = freigeben
- delegate = delegieren
- fail = fehlschlagen
- suspend = pausieren, anhalten

- resume = fortsetzen, wiederaufnehmen
- complete = abschließen
- skip = überspringen

Vor allem ist es wichtig hervorzuheben, dass „assign“ hier als eine Zuordnung von Workitems an User unabhängig vom Status definiert wird. Daher wird für jegliche Verteilung von Workitems an User unabhängig vom Status „assign“ und somit die Übersetzung „zuordnen“ genutzt. In Kontexten, in denen der Status des Workitems relevant ist, werden, entsprechend des Status, bewusst die Übersetzungen von „offer“ also „anbeiten“ und „allocate“ also „zuteilen“ verwendet.

## 2.2 Definitionen, Einordnung und relevante Modellierungselemente von BPMN

Die Modellierungssprache BPMN wurde ursprünglich 2003 als Standard für die Erfassung von Geschäftsprozessen veröffentlicht [93]. Die Abkürzung BPMN stand damals für Business Process Modeling Notation [93, 109]. Mit der BPMN-Notation sollten komplexe Prozesse graphisch abgebildet werden können, sodass diese sowohl für technische als auch für nicht-technische User intuitiv verständlich sind [93]. Auch in der neuen Version BPMN 2.0, ist das Ziel Prozesse standardisiert und leicht verständlich graphisch darzustellen, damit sie von sämtlichen beteiligten Personen verstanden werden können [85]. Auf diese Weise soll die Lücke zwischen dem Design und der Implementierung von Geschäftsprozessen geschlossen werden [85]. In der neuen Version BPMN 2.0 wird die ursprüngliche Notation um ein umfangreicheres zugrundeliegendes Metamodell ergänzt, wodurch Modelle, die mit BPMN 2.0 modelliert werden, nun die Möglichkeit haben, direkt ausgeführt zu werden [93, 109]. Daher wurde auch der Name aktualisiert und in der Version 2.0 steht BPMN für Business Process Model and Notation [85, 93, 109]. In der ursprünglichen BPMN Version musste der Prozessdesigner die BPMN-Prozessmodelle zuerst auf ein zwischengeschaltetes Ausführungsmodell wie WS-Business Process Execution Language (BPEL) abbilden, um sie ausführen zu können [93]. Durch BPMN 2.0 ist dies nicht mehr erforderlich und WfMS können BPMN-Modelle direkt ausführen [93]. Heutzutage hat sich BPMN als Industriestandard für die Modellierung von Prozessen etabliert [108, 109].

Das BPMN-Poster [21] ist eine komplette Übersicht sämtlicher Modellierungselemente die BPMN 2.0 zur Verfügung stellt. Für die vorliegende Arbeit sind vor allem die BPMN-Elemente: Pools bzw. Swimlanes, User Tasks, Sequence Flows, Exclusive Gateways und Events bzw. Timer Events relevant. Durch Pools und Swimlanes werden die Verantwortlichkeiten für die Tasks modelliert [21, 109]. Ein Pool kann dabei beliebig viele Swimlanes beinhalten [21]. Sowohl ein Pool als auch eine Swimlane kann somit z.B. eine bestimmte Organisation oder Rolle repräsentieren [21, 109]. Eine User Task stellt eine Aktivität im Workflow dar, die von einem User also einer menschlichen Ressource ausgeführt werden soll [21, 109]. Die Aktivität benötigt also eine User Interaktion, wofür typischerweise eine Applikation bzw. ein Graphical User Interface (GUI) geöffnet wird, mit der bzw. mit dem der User interagieren kann, um das Workitem einer Task abzuarbeiten [109]. Damit ein Workitem der User Task den gewünschten Usern zugeordnet werden kann, ist die Verknüpfung des Prozesses zu einem passenden Organisationsmodell notwendig [109]. Der Begriff „User“, welcher vor dem Begriff „Task“ steht, ist dabei ein Aufgabentyp, welcher den Charakter der Aufgabe beschreibt [21]. Sequence Flows verbinden die Aktivitäten und geben an, in welcher Reihenfolge der Prozess durchlaufen wird [21, 93, 109]. Sie werden durch einen durchgehenden Pfeil zwischen zwei Modellierungselementen repräsentiert [109]. Diese Modellierungselemente können Tasks, Events und Gateways sein [109]. Der erste Sequence Flow startet beim Start Event und der letzte endet beim End Event, dabei wird die Ausführungsreihenfolge der Aktivitäten im Prozess angegeben [21, 109]. Hierbei wird der Prozess sequenziell durchlaufen und das nächste Modellierungselement, auf das ein Sequence Flow zeigt, kann erst verwendet werden, wenn das vorherige, an welchem der jeweilige Sequence Flow startet, abgeschlossen wurde [93, 109]. Grundsätzlich ermöglichen es Gateways während der Ausführung alternative Pfade im Prozess zu durchlaufen [93]. Alle Gateways

können in BPMN grundsätzlich auf zwei verschiedene Arten verwendet werden: Entweder als sog. Join- oder als sog. Split-Modellierungselement [21, 93, 109]. Als Join-Element hat ein Gateway mindestens zwei Sequence Flows, die auf das Gateway zeigen und genau ein abgehenden Sequence Flow [109]. Es führt also mehrere Ausführungspfade zusammen [21, 109]. Als Split-Element hat ein Gateway genau einen eingehenden Sequence Flow und mindestens zwei abgehende [109]. Es teilt also einen Ausführungspfad in mehrere auf [21, 109]. Das Exclusive Gateway wird verwendet, um exklusives Verhalten abzubilden, da genau eine Option aus einer Menge von Alternativen ausgewählt wird [109]. Als Split-Element wählt das Exclusive Gateway genau einen ausgehenden Sequence Flow aus, an dem der Prozess fortläuft [21]. Für die Entscheidung nutzt das Exclusive Gateway eine Bedingung, die zur Entwurfszeit festgelegt wurde und zur Laufzeit überprüft wird [109]. Als Join-Element wartet das Exclusive Gateway, bis der erste Pfad der eingehenden Sequence Flows abgeschlossen ist, bevor es den ausgehenden Sequence Flow auslöst [21]. Timer Events können für verschiedene Zwecke eingesetzt werden, da sie sowohl zyklische Zeitintervalle, bestimmte Zeitpunkte, Zeitspannen oder Zeitlimits repräsentieren können [21, 109]. Daher können sie verwendet werden, um einen Prozess zu starten, eine Aktivität zu beenden oder den Prozess zu pausieren bzw. zu warten bis die nächste Aktivität ausgeführt wird [109]. In der vorliegenden Arbeit wurden sie ausschließlich verwendet, um den Prozess zu pausieren, bis eine Zeitspanne abgelaufen ist. BPMN kann durch Decision Model and Notation (DMN) für die Modellierung von Entscheidungen ergänzt werden [109]. In sog. DMN-Entscheidungstabellen, kann komplexe Entscheidungslogik mit mehreren Inputs und Outputs kompakt und formal, aber trotzdem gut lesbar, definiert werden [109].

## 2.3 Definition, Kontext und Vorstellung der Resource Patterns

Die Resource Patterns wurden von der Workflow Patterns Initiative erstellt, um die wichtigsten Konzepte und wiederkehrende Anordnungen der Resource Perspektive abzubilden [84, 93, 107]. Die Workflow Patterns Initiative ist eine Zusammenarbeit zwischen der Eindhoven University of Technology und der Queensland University of Technology, um grundlegende Konzepte für Workflow Technologien bereitzustellen [93, 107, 115]. Ein Workflow ist komplex und beinhaltet viele verschiedene Aspekte [93, 112]. Diese Aspekte müssen alle in den grundlegenden Konzepten für die Workflow Technologien berücksichtigt werden können, um einen Workflow umfassend betrachten bzw. abbilden zu können [93]. Dafür wurde eine Analyse von verschiedenen Modellierungsframeworks durchgeführt und es konnten drei Bereiche identifiziert werden, welche die Übereinstimmungen aller Frameworks repräsentieren [93]. Diese drei Bereiche, auch Perspektiven genannt, sind die Control-Flow Perspektive, die Data Perspektive und die Resource Perspektive [93, 101]. Die Control-Flow Perspektive beschreibt die Aktivitäten bzw. Tasks eines Prozesses und die Reihenfolge, in der diese abgearbeitet werden sollen [93, 101]. In der Data Perspektive finden sich die Informationen bzw. Daten wieder, die während der Ausführung eines Prozesses verwendet werden [93, 101]. Abschließend beschreibt die Resource Perspektive die Ressourcen, also Personen oder technische Instanzen, die den Prozess bzw. die einzelnen Workitems der Tasks ausführen, und deren Beziehungen miteinander [93, 101]. In jeder dieser Perspektiven gibt es sog. Workflow Patterns, welche die wichtigsten Konzepte und wiederkehrende Anforderungen eines Workflows für die jeweiligen Perspektive abbilden [93, 101, 105]. Die Resource Patterns beschreiben also welche Task von welcher Ressource übernommen wird und wie die Ressourcen zueinander in Beziehung stehen [84, 93, 101]. Es wurde sich bewusst für Patterns entschieden, um die wichtigsten Konzepte und wiederkehrende Anforderungen der Perspektive sowohl unabhängig von der Technologie als auch unabhängig von der Notation definieren zu können [84, 93, 101, 105]. Auf diese Weise haben die Patterns eine weitreichende Relevanz, da sie auf eine breite Spanne an Tools, Standard und Modellierungssprachen, mit dem Ziel Workflows umfassend abzubilden, anwendbar sind [84, 93, 105]. Die Patterns können somit genutzt werden, um die Funktionalitäten der beschriebenen Lösungen zu überprüfen und diese vergleichbar zu machen [84, 93, 101].

Der Fokus in der vorliegenden Arbeit liegt allerdings auf den Resource Patterns, weswegen auch nur diese Patterns erläutert werden. Insgesamt liegen 43 Resource Patterns vor [93, 107]. Diese sind in sieben verschiedenen Kategorien eingeteilt: Creation Patterns, Push Patterns, Pull Patterns, Detour Patterns, Auto-Start Patterns, Visibility Patterns und Multiple-Resource Patterns [84, 93]. Die detaillierten Erläuterungen sämtlicher Resource Patterns können in [93] gefunden werden. Die folgenden Erläuterungen der Kategorien und der Patterns werden aus [93] zitiert, da [93] den neusten Stand der Resource Patterns darstellt:

Die Creation Patterns gruppieren Patterns, welche Anweisungen bezüglich der Zuteilung bzw. Ausführung von Workitems nach der Erstellung beschreiben. Sie werden zur Entwurfszeit in Bezug auf eine Task festgelegt. Sie geben an wie mit Workitems nach der Erstellung aber vor der Ausführung umgegangen werden soll. Sie beschreiben also wie das Anbieten und Zuteilen ablaufen soll.

Das erste Creation Pattern ist „Direct Distribution“. Bei diesem Pattern wird zur Entwurfszeit fest angegeben, welche konkrete Ressource die Workitems einer Task zur Laufzeit übernehmen soll. Grundsätzlich bildet das Pattern die Möglichkeit ab, für eine Task einzelne Ressourcen direkt angeben zu können, unabhängig davon, ob die Workitems der Task der Ressource direkt zugeteilt werden oder nur angeboten werden. Ein Beispiel für „Direct Distribution“ ist, dass die Workitems der Task „Büro öffnen“ immer dem User „Sue“ zugeteilt werden und die Workitems von der Task „Bankgeschäft ausführen“ immer dem User „Bob“ angeboten werden. Bei dem Pattern „Role-Based Distribution“ wird zur Entwurfszeit angegeben, welchen Rollen zur Laufzeit die Workitems einer Task angeboten werden. Auf diese Weise kann das Workitem passenden Ressourcen, welche die entsprechende Rolle besitzen, angeboten werden und die Entscheidung welche Ressource das Workitem letztendlich annimmt und ausführt, erfolgt manuell durch die Ressourcen zur Laufzeit. Ein Beispiel für „Role-Based Distribution“ ist, dass die Workitems der Task „Inventarliste erstellen“ allen Usern der Rolle „Filialmitarbeiter“ angeboten werden. Die Inventarliste muss also immer von einem Filialmitarbeiter erstellt werden. Bei dem Pattern „Deferred Distribution“ kann zur Entwurfszeit angegeben werden, dass die Ressource, welche die Workitems einer Task übernehmen soll, erst zur Laufzeit identifiziert wird. Grundsätzlich wird durch dieses Pattern die Möglichkeit repräsentiert, die Auswahl der Ressource auf die Laufzeit zu verschieben, um flexibler zu sein. Auf diese Weise können die Zustände zur Laufzeit mitberücksichtigt werden. Die Identifizierung der Ressource erfolgt durch eine beliebige Form eines Auswahlmechanismus. Beispielhaft könnte in einem Workitem von der Task „Freiwilligen finden“ ein Administrator oder ein höher gestellter Mitarbeiter den Namen eines Users eintragen und dieser User muss dann die Workitems der nachfolgenden Task „Mitarbeiter schulen“ ausführen. Das Pattern „Authorization“ beschreibt die Möglichkeit Genehmigungen bzw. Privilegien für die User konfigurieren zu können, die unabhängig von der Zuteilung der Workitems ist. Auf diese Weise können bestimmte User mehr Aktionen bezüglich der Workitems ausführen. Grundsätzlich kann eine Art Sicherheitsstufe eingebaut werden, sodass nur bestimmte User Informationen sehen und bestimmte Aktionen ausführen können. Dies könnte z.B. umgesetzt sein, indem nur User mit einer bestimmten Berechtigung die Workitems der Task „Beförderungsantrag genehmigen“ delegieren dürfen oder einsehen können, auch wenn sie einem anderen User zugeteilt sind. Bei dem Pattern „Separation of Duties“ kann festgelegt werden, dass zwei Task nie von der gleichen Ressource übernommen werden dürfen. Dadurch können Prüfungskontrollen abgebildet werden. Es kann sichergestellt werden, dass z.B. die Workitems der Task „Scheck vorbereiten“ und die Workitems von „Scheck gegenzeichnen“ nicht von dem gleichen User ausgeführt werden können. Das Pattern „Case Handling“ bildet den Ansatz ab, dass alle Workitems in einem Case derselben Ressource zugeordnet werden. Die Entscheidung, welche Ressource die Workitems aus dem Case übernehmen soll, erfolgt wenn der Case oder das erste Workitem in dem Case zugeteilt werden muss. Auf diese Weise erhält z.B. derselbe Anwalt sämtliche Workitems in dem „Mandant verteidigen“-Prozess. Es wird sichergestellt, dass derselbe Anwalt sämtliche Workitems übernimmt und den Mandanten von Anfang

bis Ende des Prozesses begleitet. Dieses Pattern verfügt über zwei Versionen, welche in dem Kapitel „3.2.1 Untersuchung der Creation Patterns“ erläutert werden. Das Pattern „Retain Familiar“ ist eine flexiblere Variante des „Case Handling“-Patterns. Bei „Retain Familiar“ kann ein Workitem dem gleichen User zugeordnet werden, der ein Workitem einer beliebigen vorherigen Task im Prozess ausgeführt hat. Es sind also nicht alle Workitems der Tasks in einem Case der gleichen Ressource zugeordnet, sondern für einzelne Tasks kann festgelegt werden, dass eine Ressource die Workitems dieser Task ausführen soll, die bereits ein Workitem einer vorherigen Task im Case übernommen hat. Auf diese Weise wird die Durchlaufzeit des Prozesses verringert, da die Ressource den Case bereits kennt. Beispielhaft kann mit diesem Pattern festgelegt werden, dass der gleiche User, der die Inventur gemacht hat, auch die Bestellung aufgeben muss. Allerdings kann ein Workitem der letzten Task im Prozess „Bestellung annehmen“ von einem beliebigen Mitarbeiter ausgeführt werden. „Retain Familiar“ wurde nur für die ersten beiden Tasks im Prozess angewendet. Bei dem Pattern „Capability-Based Distribution“ werden Workitems Ressourcen zugeteilt, aufgrund von Fähigkeiten, die die Ressourcen besitzen. Diese werden im Organisationsmodell hinterlegt und können in der Capability-Function, welche den häufig programmierten Zuteilungsmechanismus abbildet, berücksichtigt werden. Auf diese Weise können für die Workitems passende Ressourcen gezielter ausgewählt werden. Als Beispiel könnten die Workitems der Task „Flugwerk prüfen“, nur Mechaniker erhalten, welche einen passenden Abschluss, die Zertifizierung für den Flugzeugtyp und mehr als zehn Jahre Berufserfahrung besitzen. Auch dieses Pattern verfügt über zwei Varianten, welche in dem Kapitel „3.2.1 Untersuchung der Creation Patterns“ vorgestellt werden. Für das Pattern „History-Based Distribution“ werden Workitems aufgrund der Ausführungshistorie der Ressource zugeordnet. Dabei werden Daten aus der Ausführungshistorie in der sog. Historical-Distribution-Function verwendet, welche den Zuteilungsmechanismus darstellt. Dieser wird häufig durch Code-Abfragen umgesetzt. Auf diese Weise werden bei der Ressourcenauswahl Daten, die äquivalent zu Erfahrungsberichten sind, berücksichtigt. Es können beliebige Daten aus der Ausführungshistorie verwendet werden, beispielhaft welcher Arzt die Task „Herz OP“ am häufigsten übernommen hat oder welcher Arzt die geringste Anzahl an fehlgeschlagenen OP-Workitems hat. Bei dem Pattern „Organisational Distribution“ werden die Workitems Ressourcen zugeordnet, aufgrund der Positionen und Beziehungen der Ressourcen innerhalb der Organisation, durch die sog. Organizational-Distribution-Function. Hierfür muss eine Möglichkeit vorliegen die Organisationsstrukturen in einem Organisationsmodell abzubilden. Zum Beispiel dürfen die Workitems der Task „Audit überprüfen“ nur dem User zugeordnet werden, der die Position CFO der Organisation belegt. Das letzte Creation Pattern „Automatic Execution“ unterschiedet sich von den anderen, da dieses Pattern das Verhalten abbildet, dass ein Workitem automatisch ausgeführt wird, ohne die Interaktion oder Zuteilung zu einer Ressource zu benötigen. Diese Workitems werden automatisch ausgeführt, wenn gewisse Kriterien erfüllt sind bzw. der Prozess an der entsprechenden Task angelangt ist. Häufig werden Interaktionen mit Systemen oder Skripts ausgeführt, wie z.B. das eine ID automatisch ermittelt wird oder ein Backup durchgeführt wird.

Die Push Patterns charakterisieren Situationen, in denen neu erstellte Workitems proaktiv durch das System den Ressourcen angeboten oder zugeteilt werden. Sowohl beim Anbieten der Workitems als auch bei der direkten Zuteilung geht die Initiative vom System aus.

Das erste Push Pattern ist „Distribution by Offer - Single Resource“. Hier bietet das System das Workitem einer einzelnen Ressource an. Dies ist nicht verpflichtend und ist äquivalent dazu jemandem darum zu bitten die Bearbeitung des Workitems in Erwägung zu ziehen. Grundlegend beschreibt das Pattern die Möglichkeit, ein Workitem gezielt einer Ressource anzubieten. Auf diese Weise können z.B. die Workitems der Task „Beweise sammeln“ dem User „Bob“ angeboten werden und die Workitems der Task „Verteidigung vorbereiten“ dem User „Sue“. Das Pattern „Distribution by Offer - Multiple Resources“ verhält sich ähnlich wie das vorherige mit dem Unterschied, dass das System

das Workitem mehreren Ressourcen anbietet. Hier kann das Workitem sowohl mehreren Usern einzeln oder einer Gruppe angeboten werden. Damit beschreibt das Pattern das Verhalten nach einem Freiwilligen aus einer Gruppe zu fragen. Beispielhaft können die Workitems der Task „Beweise sammeln“ den Usern „Bob“ und „Sue“ angeboten werden und die Workitems der Task „Verteidigung vorbereiten“ Usern, welche die Rolle „Anwalt“ besitzen. Bei dem Pattern „Distribution by Allocation - Single Resource“ wird ein Workitem einem User direkt verbindlich zugeteilt. Dies ist äquivalent zu dem Verhalten einen Verantwortlichen bzw. Besitzer zu bestimmen. Zum Beispiel können die Workitems der Task „Jahresbericht vorbereiten“ direkt dem User „Bob“ und die Workitems der Task „Jahresbericht genehmigen“ dem User „Sue“ zugeteilt werden. Das Pattern „Random Allocation“ beschreibt die Möglichkeit Workitems einem aus einer Gruppe an potenziellen User zufällig bestimmten User zuzuordnen. Das Pattern bietet die Möglichkeit eine Gruppe an potenziellen Usern anzugeben, aus denen dann zufällig zur Laufzeit einer bestimmt wird, der das Workitem übernimmt. Dies kann genutzt werden, wenn es nicht relevant ist, welche konkrete Ressource ein Workitem übernimmt. Ein Beispiel hierfür ist ein Workitem der Task „Fall beurteilen“, für das ein zufälliger Richter bestimmt wird, da es von jedem beliebigen Richter übernommen werden könnte. Bei „Round Robin Allocation“ wird eine Ressource aus einer Gruppe an potenziellen Ressourcen auf zyklischer Basis bestimmt. Auf diese Weise soll es eine faire Verteilung der Workitems auf die einzelnen potenziellen Ressourcen geben. Jede Ressource soll mit der Zeit gleich viele Workitems übernehmen. Dies kann auf verschiedenen Weisen umgesetzt werden, z.B. indem die Ressource gewählt wird, die am längsten kein Workitem dieser Task übernommen hat oder die Ressource, welche am wenigsten Workitems dieser Task übernommen hat. Falls z.B. die Fälle zwischen den Richtern gleich bzw. fair aufgeteilt werden sollen, würde sich dieses Pattern anbieten. Für das Pattern „Shortest Queue“ wird die Ressource aus einer Menge an potenziellen Ressourcen ausgewählt, welche die kürzeste Workqueue hat. Die Ressource, welche am wenigsten ausgelastet ist, da ihr Arbeitsvorrat am wenigsten Workitems beinhaltet, soll das nächste Workitem übernehmen. Auf diese Weise soll die Durchlaufzeit des Prozesses verringert werden, da die Ressource das neue Workitem potenziell am schnellsten ausführen sollte. Sollte als Beispiel der Richter „Shaun“ zwei Workitems, der Richter „Nathan“ ein Workitem und die Richterin „Alicia“ kein Workitem auf der Worklist haben, so wird das neue Workitem „Alicia“ zugeteilt, da diese es am schnellsten bzw. direkt abarbeiten könnte. Bei dem Pattern „Early Distribution“ werden die Workitems angekündigt und potenziell zugeordnet, bevor sie tatsächlich aktiviert werden. Dies wird genutzt, damit Ressourcen bereits im Vorhinein bestätigen können, dass sie das Workitem übernehmen, sodass es zu keinen Wartezeiten im Prozess für die Ressourcenzuteilung kommt und die Durchlaufzeit optimiert wird. Die Workitems werden häufig zu Beginn des Prozesses angekündigt sind aber erst ausführbar, wenn der Prozess an der entsprechenden Task angekommen ist. Ein Beispiel dafür ist, dass der Pilot „Chris“ einen Monat im Vorhinein über das Workitem für den Flug nach London benachrichtigt wird, sodass er angeben kann, ob er den Flug übernehmen kann oder ein anderer Pilot gesucht werden muss. Das Pattern „Distribution on Enablement“ benachrichtigt und verteilt die Workitems zu dem Zeitpunkt, an dem der Prozess an der entsprechenden Task angekommen ist. Das Pattern bildet somit die Standardmöglichkeit ab, Workitems zuzuordnen. Sobald die Task aktiviert wird, indem der Prozess dort angelangt ist, wird ein Workitem erstellt und angeboten bzw. zugeteilt, sodass das Workitem sofort bereit ist von der entsprechenden Ressource ausgeführt zu werden. Als Beispiel wird das Workitem der Task „Zeitung austragen“ direkt dem User „Bob“ zugeordnet, sobald der Prozess an der Task angelangt ist und „Bob“ kann sofort damit beginnen die Zeitung auszutragen. Bei „Late Distribution“ wird eine Ressource erst über ein Workitem benachrichtig bzw. erhält dieses, nachdem der Prozess bereits an der entsprechenden Task angekommen ist. Dies kann potenziell deutlich später als die Aktivierung der Task sein. Auf diese Weise soll der Prozess Nachfrageorientiert sein und die Anzahl an aktiven Workitems im Prozess verringern, sodass der User nicht überfordert wird von einem zu hohen Arbeitsvorrat. Hierbei können mehr als nur zeitliche Faktoren berücksichtigt werden, auch inhaltliche Faktoren sind möglich. So könnte ein

Workitem der Task „Auto reparieren“ z.B. erst angekündigt bzw. zugeteilt werden, wenn die Workqueue der Ressource weniger als drei Workitems beinhaltet.

Die Pull Patterns gruppieren die Patterns, welche sich auf Situationen beziehen, in denen einzelne Ressourcen die Initiative ergreifen, um Workitems zu identifizieren und sich zu deren Durchführung zu verpflichten. Die Pull Patterns werden dabei grundsätzlich von der Ressource aus initiiert.

Das erste Pull Pattern ist „Resource-Initiated Allocation“. Hier kann sich eine Ressource für die Durchführung eines Workitems verpflichten, ohne direkt mit der Arbeit an dem Workitem starten zu müssen. Nachdem das Workitem der Ressource zugeteilt ist, ist sie für die Ausführung des Workitems verantwortlich. Durch das Pattern kann eine Ressource signalisieren, dass sie das Workitem in der Zukunft ausführen möchte. Dabei kann das Workitem vorher nur einer Ressource oder mehreren Ressourcen angeboten worden sein. Wichtig ist, dass das Workitem den Status von „offered“ auf „allocated“ ändert und nur noch die Ressource daran arbeiten kann, die es angenommen hat bzw. der das Workitem also zugeordnet ist. Beispielhaft kann ein Buchhalter ein Workitem der Task „Audit durchführen“ annehmen, um sich für die Durchführung zu verpflichten. Dieses Workitem kann entweder nur ihm oder allen Buchhaltern angeboten worden sein, aber nachdem er das Workitem angenommen hat, kann nur noch er damit interagieren. Das Pattern „Resource-Initiated Execution - Allocated Work Item“ beschreibt die Möglichkeit das eine Ressource die Arbeit an einem ihr zugeteilten Workitem starten kann. Dabei wird der Status des Workitems von „allocated“ auf „started“ geändert. Ein Anwalt kann z.B. das ihm zugeteilte Workitem der Task „Beweise sammeln“ aus diesem Status heraus starten, um daran zu arbeiten. Bei „Resource-Initiated Execution - Offered Work Item“ kann eine Ressource ein ihr angebotenes Workitem auswählen und sofort mit der Arbeit daran starten, ohne es vorher annehmen zu müssen. Der Status des Workitem wechselt somit von „offered“ direkt zu „started“. Damit kann nur noch die Ressource das Workitem bearbeiten, die es auch gestartet hat, unabhängig davon, ob das Workitem einer Ressource oder mehreren Ressourcen angeboten wurde. Auf diese Weise soll die Durchlaufzeit des Prozesses verbessert werden, indem die Allokation, also die Angabe, dass die Ressource das Workitem in der Zukunft ausführen wird, ausgelassen wird und die Ressource stattdessen sofort mit der Arbeit beginnt. Ein Workitem der Task „Beweise sammeln“ wird beispielhaft so lange allen Anwälten angeboten, bis einer das Workitem ausführen kann. Dieser startet das Workitem sofort aus dem „offered“-Status heraus, ohne es vorher annehmen zu müssen und nur dieser Anwalt kann danach noch mit dem Workitem interagieren. Das Pattern „System-Determined Work Queue Content“ beschreibt die Fähigkeit des Systems, die Reihenfolge und den Inhalt anzugeben, wie die Workitems der Ressource präsentiert werden. Auf diese Weise soll das System beeinflussen können welche Workitems in welcher Reihenfolge abgearbeitet werden. Dabei kann die Standardsortierung für die Reihenfolge angepasst werden und es können Filter festgelegt werden, welche Workitems angezeigt werden, für die inhaltliche Komponente. Die Filter und Sortierungen können entweder pro User festgelegt werden oder für einen gesamten Prozess. Das System kann z.B. nur Workitems anzeigen, die eine Priorität von „high“ haben oder die Workitems nach Fälligkeitsdatum sortieren. Das Pattern „Resource-Determined Work Queue Content“ verläuft nach dem gleichen Prinzip wie „System-Determined Work Queue Content“, mit dem Unterschied, dass hier die Ressource die Reihenfolge und den Inhalt der Workitems selbst bestimmen kann. Auf diese Weise sind die Ressourcen flexibler darin welche Workitems sie wann bearbeiten und individuelle Workitems zu finden, die sie bearbeiten wollen. Eine Ressource kann die Sortierung und Filter auf seiner eigenen Worklist anpassen und ggf. auch persistent abspeichern für erneute Verwendung. Beispielhaft kann der User einen Filter einstellen, sodass ihm nur Workitems derselben Task angezeigt werden oder er kann die Worklist nach Erstellungsdatum sortieren. Das letzte Pull Pattern „Selection Autonomy“ beschreibt die Möglichkeit, dass Ressourcen selbstständig wählen können, welche Workitems sie ausführen wollen. Diese Entscheidung kann eigenständig aufgrund der Charakteristiken des Workitems oder der eigenen Präferenz getroffen werden. Auf diese Weise sollen die

Ressourcen ermächtigt werden und flexible vorgehen können. Eine Ressource die z.B. drei Workitems auf ihrer Worklist hat, kann mit dem Pattern „Selection Autonomy“ auswählen welches dieser Workitems sie ausführen will, während sie ohne „Selection Autonomy“ vom System vorgegeben bekommt, welches Workitem sie als nächstes abarbeiten muss.

Die Detour Patterns beschreiben verschiedene Möglichkeiten, wie die Zuordnung und der Lebenszyklus von Workitems von den Richtlinien, die für sie zur Entwurfszeit festgelegt wurden, abweichen können.

Das erste Detour Pattern ist „Delegation“. Bei diesem Pattern kann eine Ressource ein ihr zugeordnetes aber noch nicht gestartetes Workitem einer anderen Ressource zuordnen. Auf diese Weise erhält die Ressource die Möglichkeit Workitems abzugeben, da sie diese momentan nicht ausführen kann oder nicht ausführen will. Nachdem das Workitem delegiert wurde, kann nur noch die neue Ressource darauf zugreifen. Beispielhaft könnte der User „Bob“ ein Workitem der Task „Audit prüfen“ an „Sue“ delegieren, bevor er Urlaub hat. Für das Pattern „Escalation“ führt das System die neue Zuordnung eines Workitems aus. Dieses kann sich in verschiedenen Zuständen befinden und neuen Ressourcen in beliebigen Zuständen zugeordnet werden. Eine Übersicht sämtlicher Zustandstransfers dieses Pattern lässt sich in der Abbildung 2 finden. Das System ordnet das Workitem neuen Ressourcen zu, um die Fertigstellung des Workitems zu beschleunigen. Auslöser für die neue Zuordnung kann das Ablaufen eines Fälligkeitsdatums sein oder auch um die Last des Arbeitsvorrats vorbeugend auszugleichen. Die neue Zuordnung wird entweder automatisch durch das System oder manuell durch einen Administrator ausgeführt. Dabei wird das neu zugeordnete Workitem nicht wieder der ursprünglichen Ressource zugeordnet. Ein Beispiel für das Pattern wäre, dass ein Workitem der Task „Zahlung prüfen“ dem Buchhalter „Max“ zugeteilt ist und das Workitem, nachdem das Fälligkeitsdatums abgelaufen ist, durch das System den restlichen Buchhaltern angeboten wird. Bei dem Pattern „Deallocation“ kann eine Ressource ein ihr zugeteiltes Workitem, welches noch nicht gestartet wurde, wieder freigeben, sodass es einer anderen Ressource oder einer Gruppe an Ressourcen zugeordnet werden kann. Auf diese Weise hat die Ressource die Möglichkeit ein Workitem wieder abzugeben, sodass es den anderen Ressourcen wieder zur Verfügung gestellt und neu zugeordnet wird. Ein Buchhalter könnte z.B. sein „Zahlung prüfen“ Workitem wieder freigeben, wenn er keinen Fortschritt macht, wenn ein passenderer Buchhalter frei ist oder wenn er zu viele Workitems besitzt. Das Pattern „Stateful Reallocation“ beschreibt die Fähigkeit einer Ressource ein Workitem, welches sie gerade ausführt einer anderen Ressource zuzuteilen ohne den Fortschritt also die eingegeben Daten zu verlieren. Auf diese Weise kann eine Ressource Arbeit abgeben, ohne den bisherigen Fortschritt zu verlieren. Dieses Pattern ist ähnlich zu „Delegation“, allerdings kann „Delegation“ nur Workitems neu zuordnen, die noch nicht bearbeitet wurden, während „Stateful Reallocation“ Workitems neu zuordnet, die in Bearbeitung sind. Die Eingaben bzw. Daten mit denen das Workitem bereits befüllt wurde, bleiben der neuen Ressource erhalten. Ein Buchhalter kann durch das Pattern z.B. das halbe Formular des Workitems „Zahlung prüfen“ bereits ausgefüllt haben und ordnet es für die zweite Hälfte einem anderen Buchhalter für die Fertigstellung des Workitems zu. Das Pattern „Stateless Reallocation“ stellt die Option dar, dass Ressourcen Workitems, die sie gerade bearbeiten, einer anderen Ressource zuteilen können, ohne dass der Fortschritt gespeichert wird. Dies bildet eine einfacher umzusetzende Variante von „Stateful Reallocation“ ab, da die Komplexität entfällt, die bereits eingegeben Daten zu speichern und mitzusenden. Der Fortschritt und die eingegebenen Daten gehen verloren und die neue Ressource muss bei der Bearbeitung von vorne anfangen. Beispielhaft kann der Fortschritt eines Buchhalters beim Workitem „Zahlung prüfen“ nicht ausreichend voranschreiten, weswegen er es einem anderen Buchhalter zuordnet, welcher bei der Prüfung der Zahlungen von vorne startet. Für das Pattern „Suspension-Resumption“ benötigt eine Ressource die Fähigkeit die Bearbeitung von Workitems zu pausieren und wieder fortzusetzen. Auf diese Weise kann die Ressource signalisieren, dass sie die Bearbeitung temporär anhält. So kann sich die Ressource auf andere

Workitems fokussieren, bis sie die Bearbeitung des pausierten Workitems wieder aufnimmt. Die Ressource kann beide Aktionen, also das Pausieren und das Fortsetzen, initiieren und die Arbeit am Workitem zu einem beliebigen Zeitpunkt in der Zukunft fortsetzen. Gründe für das Pausieren könnten z.B. sein, dass ein Buchhalter zu einer Konferenz muss und erst danach mit der Bearbeitung des „Zahlung prüfen“ Workitems fortfahren kann. Bei dem Pattern „Skip“ kann eine Ressource ein ihr zugeteiltes Workitem überspringen, sodass es abgeschlossen ist. Auf diese Weise ist es möglich nicht kritische Workitems zu überspringen, um anschließende Workitems sofort starten zu können und so die Durchlaufzeit des Prozesses zu verringern. Dafür kann der Status des Workitems von „allocated“ auf „completed“ geändert werden, ohne dass eine Ressource die Arbeit am Workitem starten musste. Ein Anwalt könnte z.B. das Workitem für „Beweise sammeln“ abarbeiten und das Workitem für „Zeugen befragen“ überspringen, um direkt das Workitem „Verdächtigen anklagen“ auszuführen, da er bereits alle Beweise besitzt, die er benötigt. Bei „Redo“ kann eine Ressource ein Workitem wiederholen, welches es zuvor in dem Case bereits abgeschlossen hat. Sämtliche folgende Workitems müssen aus Gründen der Konsistenz auch wiederholt werden. Dabei müssen auch die eingegebenen Datenelemente überprüft und ggf. erneuert werden. Auf diese Weise kann ein Workitem erneut abgearbeitet werden, falls neue Erkenntnisse vorliegen oder Fehler festgestellt wurden. Dieses Zurückspulen des Cases auf einen früheren Zustand ist vor allem problematisch und komplex, wenn mehrere User in einem Case beteiligt sind, vor allem in Bezug auf die ausgefüllten Datenelemente oder externe Effekte außerhalb des Systems. Ein Beispiel für das Pattern wäre, ein Anwalt, der den gesamten Case betreut hat, erhält vor der Verhandlung neue Informationen, weswegen er das Workitem für „Zeugen befragen“ erneut ausführen muss, auch wenn er dieses bereits abgeschlossen hatte. Das letzte Detour Pattern ist „Pre-Do“. Hier soll es für eine Ressource möglich sein, ein Workitem vor dem Zeitpunkt auszuführen, an dem es Ressourcen angeboten oder zugeteilt wird. Dabei können nur die Workitems zuvor abgearbeitet werden, die keine Daten von vorherigen Workitems benötigen. Grundlegend kann eine Ressource auf diese Weise die Workitems in einem Case so früh wie möglich abarbeiten, ungeachtet der vorgegebenen Reihenfolge durch das Prozessmodell. Der User und nicht das Prozessmodell soll aussuchen können, in welcher Reihenfolge die Workitems abgearbeitet werden sollen, um die Durchlaufzeit des Prozesses zu verringern. Auf diese Weise kann z.B. der Anwalt bereits das Workitem für die zweite Task im Prozess „Zeugen befragen“ ausführen, bevor er ein Workitem der ersten Task im Prozess „Beweise sammeln“ abgeschlossen hat.

Die sog. Auto-Start Patterns gruppieren Patterns, welche alternative Wege aufzeigen, wie Workitems automatisch gestartet werden können. Diese alternativen Wege können z.B. durch bestimmte Ereignisse im Lebenszyklus des Workitems oder des zugehörigen Workflows ausgelöst werden.

Das erste Auto-Start Pattern ist „Commencement on Creation“. Hier kann eine Ressource direkt nach der Erstellung des Workitems mit der Arbeit daran beginnen. Die Erstellung, die Zuteilung und das Starten des Workitems erfolgt gleichzeitig durch das System, sodass die Ressource direkt anfangen kann an dem Workitem zu arbeiten. Durch das Pattern soll die Durchlaufzeit des Prozesses verringert werden, da die Zeit, bis die Ressource das Workitem annimmt und die Zeit, bevor die Ressource mit der Arbeit am Workitem beginnt, entfällt. Beispielhaft wird das Workitem der Task „Audit durchführen“ direkt im Status „started“ in die Worklist einer Ressource hinzugefügt, sodass sie direkt das Formular des Workitems bearbeiten kann. Bei dem Pattern „Commencement on Allocation“ kann die Ressource sofort mit der Arbeit an dem Workitem starten, sobald dieses ihr zugeteilt wurde, ohne es extra starten zu müssen. Das System startet das Workitem sofort, sobald eine Ressource es annimmt. Das Zuteilen und Starten wird durch das System gleichzeitig ausgeführt. Ein Workitem ändert somit direkt seinen Status von „offered“ auf „started“, nachdem es angenommen wurde. Dieses Pattern gilt als Abstufung von „Commencement on Creation“. Dadurch das ein Workitem erst gestartet wird, nachdem es zugeteilt wurde und nicht wenn es erstellt wurde, kann die Identifikation einer

passenden Ressource später im Workitem Lifecycle durchgeführt werden. Auf diese Weise werden sowohl Vorteile durch die Flexibilität der Ressourcenauswahl als auch Vorteile durch die Verringerung der Durchlaufzeit zwischen Workitem Zuteilung und Start, beibehalten. Nachdem das Workitem der Task „Audit durchführen“ mehreren Buchhaltern angeboten wurde, nimmt es „Bill“ an. Das Workitem wird automatisch durch das System nur noch „Bill“ angezeigt, und zwar im Status „started“, sodass er sofort mit der Arbeit daran beginnen kann. Das Pattern „Piled Execution“ beschreibt, dass alle Workitems einer Task nacheinander abgearbeitet werden, sodass die Ressource immer Stapel derselben Art von Workitem abarbeitet. Dies soll die Bearbeitung des Workitems optimieren, da die Ressource mit der Bearbeitung des Workitems bereits vertraut ist. Alle Workitems derselben Task werden von einer Ressource nacheinander abgearbeitet. Der Wechsel in diesem Modus erfolgt durch die Ressource und es können auch mehrere Ressourcen den Modus für die gleiche Task aktiviert haben. Auf diese Weise werden die Workitems fallübergreifend stapelweise abgearbeitet. Die Workitems werden nacheinander gestartet, sodass vorgegeben ist, welches Workitem als nächstes ausgeführt wird. Eine Putzkraft, welche die Hotelzimmer erst aufräumen und dann sauber machen soll, würde z.B. mit diesem Pattern zuerst alle Zimmer aufräumen, um sie anschließend in einem zweiten Durchgang alle sauber zu machen. Das letzte Auto-Start Pattern ist „Chained Execution“. Hier wird nacheinander jedes Workitem eines Cases durchlaufen, bis dieser Case abgeschlossen ist. Auf diese Weise soll die Durchlaufzeit des Prozesses verringert werden, da eine Ressource aufeinanderfolgende Workitems in einem Case erhält, um im Case fortzuschreiten. Diese Workitems werden nacheinander gestartet, sodass die Reihenfolge der Bearbeitung vorgegeben ist. Jedes Workitem startet automatisch seinen Nachfolger, welcher im Status „started“ der jeweiligen Ressource zugeordnet wird. Auf diese Weise werden die Workitems fallweise abgearbeitet, bis ein Case beendet wurde. Der Wechsel in diesem Modus erfolgt durch die Ressource. Beispielhaft würde eine Putzkraft, welche die Hotelzimmer erst aufräumen und dann sauber machen soll, mit diesem Pattern zuerst ein Zimmer aufräumen und sauber machen, bevor die Putzkraft zum nächsten Zimmer übergehen würde.

Die Visibility Patterns geben an inwieweit die Ressourcen ausstehende und ausgeführte Workitems beobachten können.

Das erste Visibility Pattern ist „Configurable Unallocated Work Item Visibility“. Dieses Pattern beschreibt die Möglichkeit, die Sichtbarkeit von nicht zugeteilten Workitems pro Ressource konfigurieren zu können. Hierbei ist die Möglichkeit des Systems gefragt, die Sichtbarkeit von nicht zugeteilten Workitems einschränken oder freigeben zu können. Vor allem für Workitems, die angeboten werden, ist diese Funktion interessant. Die Sichtbarkeit könnte für potenzielle oder grundsätzlich sämtliche Ressourcen eingeschränkt werden. Ein Anwendungsbeispiel wäre, dass zwei Arbeiter nur jeweils die eigenen Workitems, die ihnen angeboten werden, sehen sollen und der Vorgesetzte alle Workitems sehen soll, welche den Mitgliedern aus seinem Team angeboten werden. Das zweite und letzte Visibility Pattern ist „Configurable Allocated Work Item Visibility“. Hier wird beschrieben, dass die Sichtbarkeit von zugeteilten Workitems pro Ressource konfiguriert werden kann. Dieses Pattern verhält sich gleich wie „Configurable Unallocated Work Item Visibility“, mit dem Unterschied, dass die Sichtbarkeit von zugeordneten oder gestarteten Workitems konfiguriert werden kann. Diese Workitems können anderen Usern zugeteilt sein. Auch hier kann das Anwendungsbeispiel genutzt werden, dass die Mitarbeiter jeweils nur die eigenen ihnen zugeteilten Workitems sehen sollen, aber der Vorgesetzt sämtliche Workitems, welche den Mitgliedern aus seinem Team zugeteilt sind, sehen soll.

Die letzte Kategorie Multiple-Resource Patterns gruppiert Patterns, welche die Situationen identifizieren, in denen die Zuordnung zwischen Ressourcen und Workitems nicht eins zu eins ist.

Das erste Multiple-Resource Pattern ist „Simultaneous Execution“. Bei diesem Pattern wird eine 1-zu-N-Beziehung betrachtet, sodass eine Ressource an mehreren Workitems

gleichzeitig arbeiten kann. Hier wird die Situation abgebildet, dass nicht ein Workitem nach dem nächsten abgearbeitet werden kann, sondern die Ressource mehrere Workitems gleichzeitig bearbeitet und in der Bearbeitung zwischen den Workitems hin und her wechselt. Dieses Pattern bietet einen flexibleren Ansatz, bei dem die Auswahl, welche Workitems in welcher Kombination ausgeführt werden sollen, der Ressource überlassen und nicht vom System vorgegeben wird. Als Beispiel könnte ein Anwalt gleichzeitig die Workitems für die Tasks „Beweise sammeln“ und „Verteidigung vorbereiten“ ausführen und in der Bearbeitung beliebig zwischen ihnen wechseln. Das andere Multiple-Resource Pattern ist „Additional Resources“. Dieses Pattern betrachtet eine N-zu-1-Beziehung, sodass mehrere Ressourcen an einem Workitem arbeiten können. Das Pattern bildet die Funktion ab, dass eine Ressource bei einem Workitem, was sie aktuell bearbeitet, Unterstützung von anderen Ressourcen anfordern kann. In komplexen Kontexten benötigen Workitems häufig die Bearbeitung von mehreren Ressourcen. Auch in der Praxis unterstützen sich User häufig bei Workitems gegenseitig und bearbeiten eine Aufgabe mit mehreren Personen. Beispielweise könnte ein Buchhalter das Workitem „Audit durchführen“ bearbeiten, aber da es ein sehr komplexes Audit und eine große Aufgabe ist, benötigt er Unterstützung von anderen Buchhaltern, um das Workitem abzuschließen. Hier könnte der Buchhalter Unterstützung anfragen, sodass sie das Workitem gemeinsam fertigstellen.

## 2.4 Stand der Forschung und resultierende Relevanz der Arbeit

### 2.4.1 Vorgehen der Literaturrecherche

Für die Literaturrecherche wurde zuerst das Buch [93] identifiziert, welches die neueste Veröffentlichung der Workflow Patterns Initiative, bezüglich der Resource Patterns darstellt. Aufgrund der Beschreibungen in [93] und der Übersicht auf der offiziellen Website der Workflow Patterns Initiative [113], konnte die Quelle [84] identifiziert werden, welche die Resource Patterns initial einführt. Zusätzlich konnten die Originalquellen für die Untersuchungen der beiden in der vorliegenden Arbeit relevanten Aspekte, Open-Source-Tools [110] und BPMN [111], herausgearbeitet werden. Auf Grundlage dieser Originalquellen, konnte die Literaturrecherche ausgeführt werden, welche sich in zwei Teile aufteilt: das Snowballing der oben benannten Quellen und eine ergänzende Literatursuche per Suchstring.

Um passende Literatur zu finden, wurde als erstes mit den benannten Quellen sowohl ein Forward- als auch ein Backward-Snowballing durchgeführt. Mithilfe des Forward-Snowballing sollte sowohl festgestellt werden, ob [93] tatsächlich den neusten Stand der Resource Patterns darstellt als auch, ob und wenn ja, wofür die Resource Patterns aktuell verwendet werden. Hierfür wurde die „Zitiert von“-Funktion von Google Scholar verwendet. Durch das Backward-Snowballing sollte abgesichert werden, dass es sich um die originalen Quellen für die Untersuchungen handelt. Darüber hinaus konnten neue relevante Quellen durch das Snowballing identifiziert werden.

Die Ergebnisse wurden anschließend durch eine Literatursuche per Suchstring ergänzt. Hierfür wurden die folgenden beiden Suchstrings verwendet:

- „Resource Pattern“ AND „Workflow Management System“
- „Resource Pattern“ AND „Business Process Management System“

Die beiden Suchstrings wurden bewusst gewählt, um den aktuellen Stand der Forschung für Resource Patterns im Kontext von WfMS zu betrachten. Da weder die Control-Flow Perspektive noch die Data Perspektive für die zentrale Fragestellung der vorliegenden Arbeit relevant sind, müssen diese im Suchstring nicht berücksichtigt werden. Ein Suchstring, der ausschließlich aus „Resource Pattern“ besteht, wäre nicht zielführend gewesen, da dieser eine zu große Menge an Literatur hervorgebracht hätte. Außerdem sollten die gefundenen Quellen aus dem WfMS-Kontext stammen, weswegen sich für den Zusatz „Workflow Management System“ entschieden wurde. Da die Bezeichnung BPMS in der Literatur häufig synonym mit WfMS verwendet wird, wurde der zweite Suchstring

hinzugefügt [93, 101, 107]. Für beide Suchstrings wurde die Suchplattform Bib-Discover der Hochschul- und Kreisbibliothek Bonn-Rhein-Sieg mit folgenden Kriterien verwendet:

- „Online verfügbar“ aktiviert
- „Peer Reviewed Zeitschriften“ aktiviert
- „Suche im Volltext“ aktiviert

Auf diese Weise sollte nur wissenschaftlich anerkannte Literatur verwendet werden, die online durch den Fernzugriff der Hochschule Bonn-Rhein-Sieg eingesehen werden kann. Bib-Discover ist eine Schnittstelle für Medien und Inhalte aus lizensierten Datenbanken und deckt dabei alle relevanten Datenbanken, wie IEEE, ACM, usw. ab [81]. Eine Auflistung sämtlicher Datenbanken lässt sich unter [70] finden. Die resultierende Literatur wurde zuerst für einen allgemeinen Überblick nach Relevanz sortiert. Um den aktuellen Stand der Forschung zu betrachten, wurde anschließend die Sortierung auf „Datum-neuestes“ geändert. Hier wurden alle Quellen betrachtet, die nach 2016 erschienen und somit neuer als [93] sind. Die Workflow Patterns Initiative veröffentlichte mit [93] den neusten Stand der Resource Patterns, daher wurde das Erscheinungsjahr der Quelle als Grenze gewählt, um zu betrachten, wie die spätere Literatur Resource Patterns verwendet. Auf diese Weise sollte überprüft werden, ob der neueste Stand der Resource Patterns aus [93] noch aktuell ist, oder ob es bereits neuere Ergebnisse gibt. Zusätzlich wurde die Literaturrecherche durch eine Suche mit der Suchplattform Google Scholar ergänzt, um eine noch breitere Menge an Quellen zu sichten. Google Scholar wurde aufgrund der großen Reichweite an Quellen gewählt. Dies ergänzt die Ergebnisse aus Bib-Discover, um sicherzustellen, dass keine relevante Quelle übersehen wurde. Für einen Überblick über die aktuelle Literatur wurde auch bei Google Scholar alle Quellen betrachtet, die nach 2016 erschienen sind. Mithilfe der durch die Literaturrecherche identifizierten Quellen, wird im Folgenden der Stand der Forschung für Resource Patterns im Kontext von WfMS erläutert.

### 2.4.2 Resource Patterns und deren Kontext in Modellierungssprachen und WfMS

Die Resource Patterns wurden 2004 in [84] eingeführt. Eine genauere Erläuterung der einzelnen Patterns kann in dem Kapitel „2.3 Definition, Kontext und Vorstellung der Resource Patterns“ gefunden werden. Die Quelle [84] beschreibt die einzelnen Resource Patterns und untersucht, inwiefern diese in damaligen kommerziellen WfMS abgebildet werden können. Zusätzlich ist 2005 das Paper [95] erschienen, welches ausgewählte Patterns und die wichtigsten Ergebnisse von [84] zusammenfasst. 2006 erscheint das Paper [111], welches die Modellierungssprache BPMN mithilfe der Resource Patterns analysierte. Darüber hinaus wurden in [111] die Ergebnisse von Untersuchungen der Resource Patterns bei der Modellierungssprache Unified Modeling Language (UML) 2.0 Activity Diagrams [94] und dem Tool Oracle BPEL Process Manager [83] dargestellt. Der Fokus von [111] liegt aber auf BPMN. Anschließend erschien 2007 die Doktorarbeit von Russel [92], welche alle bisher veröffentlichten Untersuchungen zusammenfasst. Die Ergebnisse von [92] für die kommerziellen WfMS stimmen mit denen aus [84] überein. Die Ergebnisse für UML, Oracle BPEL und BPMN unterscheiden sich von [83, 94, 111], wodurch [92] den Ergebnisstand der Quellen erneuert. Der neuste Stand der Resource Patterns wurde 2016 in [93] veröffentlicht. Diese sind nahezu identisch zu den Patterns aus 2004, mit Ausnahme einzelner Patterns, bei denen die Namen und Beschreibungen angepasst wurden. Die Logik der einzelnen Resource Patterns ist überall gleich geblieben. Auch in [93] werden die Resource Patterns einzeln erläutert, allerdings werden andere Systeme und Modellierungssprachen betrachtet. Die Quelle [93] analysiert mithilfe der Resource Patterns die Modellierungssprache BPMN 2.0 und zwei der damals führenden WfMS BPMone und Oracle BPEL. Somit erneuert [93] die Ergebnisse für BPMN aus [92]. Außerdem betrachtet [93] die neuere Oracle BPEL Version 11, während [92] die Version 10.1.2 betrachtet. Auch untersucht [93] die Version 2.0 von BPMN, wodurch einige neue Ergebnisse entstammen. Bis auf zwei Ausnahmen, bleibt die Umsetzung der Resource Patterns in BPMN 2.0 gleich oder hat sich verbessert. „Commencement on Creation“ und „Chained Execution“ sollen laut [111] und [92] in BPMN möglich sein,

sind aber in BPMN 2.0 nach [93] nicht mehr abbildbar. Mithilfe der Resource Patterns wurden neben kommerziellen WfMS in [84] auch Open-Source-WfMS in [110] untersucht. [110] analysiert die drei Open-Source-Tools jBPM, OpenWFE und Enhydra Shark und fasst die Ergebnisse zusammen. Bis auf [83] handelt es sich bei den bisherigen Quellen um Originalquellen bzw. -untersuchungen der Workflow Patterns Initiative, geführt von den Professoren van der Aalst und ter Hofstede.

Die Ergebnisse aller bisherigen Quellen zeigen, dass sowohl die Modellierungssprachen als auch die WfMS die Resource Patterns und somit auch die Resource Perspektive nicht ausreichend abbilden können [84, 93, 110, 111]. Diese Aussage lässt sich auch in den zusätzlichen Quellen wiederfinden, welche durch das oben beschriebene Vorgehen identifiziert wurden [23, 69, 72, 74, 77, 78, 86, 97]. Diese besagen, dass die Modellierungssprachen die Resource Perspektive nicht ausreichend abbilden und somit die Resource Patterns nicht umgesetzt werden können. Die folgenden Quellen [23, 69, 74, 77, 78, 97] beziehen sich hierbei auf BPMN und zitieren dafür die Originalquellen der Workflow Patterns Initiative, während [72, 86] dieselbe Aussage für andere Modellierungssprachen, unter anderem auch für BPEL treffen. Die Aussage wird weiter unterstrichen, da auch die Modellierungssprache Case Management Model and Notation (CMMN) fast keine Resource Patterns nativ abbilden konnte [56]. Dies ist vor allem bemerkenswert, da sich BPMN als Industriestandard für Prozessmodellierung etabliert hat [108, 109] und somit in aktuellen WfMS genutzt wird. An den Ergebnissen von [84, 93, 110] lässt sich erkennen, dass die untersuchten WfMS einen Großteil der Resource Patterns nicht umsetzen konnten. Vor allem der Vergleich zu den Ergebnissen der anderen Perspektiven bzw. Patterns unterstreicht die Aussage, dass der Fokus der Modellierungssprachen und WfMS eher auf die Control-Flow und Data Perspektive gelegt wurde [93]. Auch [101] benennt die fehlende Unterstützung für die Resource Perspektive in WfMS explizit. Aufgrund der Fülle an Quellen und der Erscheinungsdaten von [23, 72, 78, 97] zwischen 2017 und 2024, lässt sich feststellen, dass diese Vernachlässigung der Resource Perspektive und Patterns in Modellierungssprachen immer noch aktuell ist, aber keine neuen Untersuchungen für WfMS vorliegen. Eine Untersuchung in diesem Bereich ist daher auch heutzutage noch relevant. Auch vor dem Hintergrund, dass BPMN als Industriestandard diese Defizite bezüglich der Resource Perspektive aufweist, erscheint eine Untersuchung von BPMN basierten WfMS sinnvoll.

Obwohl die Resource Patterns bereits 2004 eingeführt wurden, werden sie heutzutage immer noch verwendet, sowohl als Grundlage für neue Konzepte als auch zur Validierung und zum Vergleich dieser Konzepte, wie im folgenden Abschnitt erläutert. Dies lässt sich an den herangezogenen Quellen für diesen Abschnitt erkennen, welche alle 2016 oder später erschienen und somit neuer als [93] sind. Die Quellen [74, 91] nutzen die Resource Patterns, um neu erstellte Modellierungssprachen zu validieren und mit anderen zu vergleichen. Darüber hinaus belegt [68], dass die Resource Patterns als Benchmarking für Modellierungskonzepte verwendet werden. Die grundlegende Funktion der Resource Patterns als Vergleichsmöglichkeit für WfMS wird 2021 durch [79] belegt. Auch [76] nutz die Patterns, um die Funktionalitäten ihres neu erstelltes Cloud-WfMS darzustellen und so das System zu validieren und vergleichbar mit anderen WfMS zu machen. Die Quelle [22] bestätigt, dass die Resource Patterns ein etabliertes Konzept sind, um sowohl WfMS als auch Modellierungssprachen zu bewerten. Allerdings übt [22] auch leichte Kritik an den Resource Patterns, da diese in einer Zeit entstanden sind, in der es untypisch war, dass mehrere Ressourcen an einer Aufgabe gearbeitet haben. Demnach ist nur ein Pattern vorhanden, welches dieses Verhalten behandelt [22]. Des Weiteren werden die Resource Patterns in der Literatur auch als Grundlage für die Erarbeitung neuer Resource Patterns verwendet [57, 66, 72, 100]. Viele der aktuellen Quellen, befassen sich damit neue Resource Patterns für die Resource Perspektive zu erstellen und nutzten die bekannten Resource Patterns als Grundlage und/oder Validierung für die neuen Patterns [57, 66, 72, 100]. Die meisten des neu erstellten Patterns haben keine weiteren Nennungen in anderen Quellen, mit Ausnahme der Patterns aus [100], welche anstatt der originalen Resource Patterns in [72] verwendet wurden. Die Resource

Patterns aus [93] bleiben allerdings Standard für die Resource Perspektive in WfMS und werden, wie in diesem Abschnitt gezeigt, weiterhin in der Literatur verwendet. Darüber hinaus werden die Resource Patterns auch zur Validierung bzw. Überprüfung in neuen Kontexten verwendet [60, 96]. In [96] werden die Resource Patterns genutzt, um die Ausdruckskraft der Ergebnisse verschiedener Process-Mining-Ansätze zu überprüfen und in [60] wird ein Modell im Bereich der Smart-Contracts durch die Patterns validiert. Für die Quellen [2, 67, 73, 87, 109] bilden Resource Patterns die allgemeine Grundlage für neue Ideen. Diese reichen von neuen Ansätzen für die Allokation von menschlichen Ressourcen in WfMS [2] bis hin zu neuen Frameworks für die Überprüfungen der Resource Perspektive [101] oder der Job-Qualität einer Ressource in WfMS [73]. Wie in diesem Abschnitt dargestellt, wird die Relevanz des Konzeptes der Resource Patterns durch die Verwendung in aktueller Literatur unterstrichen. Dementsprechend ist es sinnvoll die Resource Patterns auch bei aktuellen Untersuchungen von WfMS zu einzusetzen.

Die restlichen Quellen, welche durch das oben benannte Vorgehen identifiziert wurden, benennen die Resource Patterns entweder nur kurz, aber verwenden sie nicht aktiv, wie z.B. in [1, 59, 63, 65, 68, 75, 82, 88] oder benennen die Resource Patterns nicht und beziehen sich häufig auf einen gänzlich anderen Kontext, wie z.B. in [64, 90, 106]. In den erstgenannten Quellen [1, 59, 63, 65, 68, 75, 82, 88] erfolgt häufig ein einzelner Verweis auf das Konzept der Resource Patterns, allerdings werden diese im weiteren Verlauf der Quelle nicht erneut erwähnt oder verwendet. Viele der Quellen wie [64, 90, 106] wurden durch das oben beschriebene Snowballing identifiziert, aber zitieren häufig andere, für den Kontext der Resource Patterns unrelevante, Aspekte der Originalquellen. Die Quellen in diesem Absatz werden beispielhaft genannt, da aufgrund der Menge an Literatur in dieser Kategorie, nicht alle aufgelistet werden können.

Demnach wurden keine Quellen identifiziert, welche die Ergebnisse aus [84, 93, 110] erneuern. Dies wird unterstrichen durch die Tatsache, dass die aktuellen Quellen, weiterhin auf die mittlerweile über 15 Jahre alten Ergebnisse von [84, 110] verweisen. Es liegen somit keine Untersuchungen der Resource Perspektive in aktuellen WfMS vor. Wie in den vorherigen Abschnitten beschrieben, besteht weiterhin das Problem, dass die Resource Perspektive in der Modellierungssprache BPMN unterrepräsentiert ist. Auch das Konzept der Resource Patterns wird aktuell verwendet und ist relevant. Aufgrund dessen ist es sinnvoll für aktuelle Ergebnisse moderne BPMN basierte WfMS mithilfe der Resource Patterns zu analysieren. Nach dem besten Wissen des Autors, existiert keine Untersuchung, welche BPMN basierte WfMS in Bezug auf die Resource Patterns analysiert und vergleicht. Des Weiteren existieren seit [84, 93, 110] keine neuen Ergebnisse und/oder Untersuchungen in diesem Bereich.

## 2.5 Auswahl und Überblick der untersuchten Tools

In der vorliegenden Arbeit werden die folgenden Tools untersucht:

- Axon Ivy (Axon Ivy Designer 11.2.1)
- Bonitasoft (Bonita Studio Community Edition 2023.2)
- Camunda (Camunda 7 Community (Version 7.21))

Die Tools wurden aus dem neusten Report von „The Forrester Wave: Digital Process Automation Software, Q4 2023“ ausgewählt [80]. Der „The Forrester Wave“-Report, ist ein Leitfaden für Kunden bei der Tool- bzw. Systemauswahl in verschiedenen technologischen Bereichen [61, 62]. Seit mehr als 18 Jahren werden die „The Forrester Wave“-Bewertungen mithilfe transparenter Methodik erarbeitet, um Kunden bei der Anbieterauswahl zu unterstützen [61, 62]. Durch den Report können Kunden die aktuellen Anbieter, deren Einflüsse und ihre Wettbewerbsposition verstehen [61]. Die Relevanz des Reports wird auch dadurch unterstrichen, dass alle drei Tools die Ergebnisse des Reports auf ihren eigenen Webseiten nutzen [19, 24, 50]. Für die vorliegende Arbeit wurde der „The Forrester Wave“-Report über Digital Process Automation Software betrachtet, da keine aktuellen Reports über WfMS oder BPMS existieren und die Definition von

Forrester für Digital Process Automation Software die betrachteten WfMS-Funktionalitäten, Prozesse zu modellieren und auszuführen, enthält [50]. Zusätzlich besagt Forrester, dass Digital Process Automation Software klassische BPM-Software beinhaltet [102]. Dementsprechend wurde der „The Forrester Wave“-Report über Digital Process Automation Software als am relevantesten eingestuft. Durch die Toolauswahl aus dem Report soll somit die Aktualität und Relevanz der Tools sichergestellt werden.

Hierbei werden nur die Tools betrachtet, die die folgenden Kriterien erfüllen:

- Der Fokus des Tools ist Workflow Management
- Das Tool verwendet BPMN
- Das Tool wird in einer Open-Source- oder kostenlosen Variante zur Verfügung gestellt

Der Fokus des Tools sollte Workflow Management sein, da in der vorliegenden Arbeit mithilfe der Resource Patterns WfMS betrachtet werden sollen. Die Patterns beziehen sich wie bereits in dem Kapitel „2.1 Definitionen, Funktionsweise und Kontext eines WfMS“ erläutert, auf die klassischen WfMS-Funktionen, also das Modellieren und Ausführen von Prozessen bzw. Workflows. Tools aus dem „The Forrester Wave“-Report über Digital Process Automation Software können über weitreichendere Funktionen verfügen oder einen anderen Fokus haben, da BPM-Software nur ein Teil des gesamten Digital-Process-Automation-Tools darstellt [102]. Durch den Fokus auf Workflow Management sollte sichergestellt werden, dass nur die relevantesten Tools betrachtet werden, bei denen die benötigten und zu analysierenden klassischen WfMS-Funktionen im Fokus stehen. BPMN wurde ausgewählt, da es sich als Industriestandard für die Modellierung von Prozessen etabliert hat und trotzdem Defizite bezüglich der Resource Perspektive aufweist [93, 108, 109]. Das letzte Kriterium nur Open-Source-Tools bzw. Tools mit einer kostenlosen Variante zu betrachten, wurde aus Gründen der Verfügbarkeit der Tools gewählt. Zusätzlich bietet dieser Ansatz Vorteile bezüglich der Vergleichbarkeit der Tools untereinander, da immer dieselbe Art von Tool verglichen wird. Darüber hinaus wird der gleiche Fokus gesetzt wie in [110], wodurch die Ergebnisse der vorliegenden Arbeit als eine Erneuerung der Ergebnisse aus [110] betrachtet werden können. Allerdings werden nicht die ursprünglichen Tools aus [110] untersucht, da diese mittlerweile nicht mehr relevant sind. Weder OpenWFE noch Enhydra Shark oder die dahinterstehenden Organisationen haben noch offizielle Webseiten und die neusten Downloadmöglichkeiten verfügen nur über veraltete Varianten von 2013 [98] bzw. 2018 [99]. jBPM wird heutzutage zwar noch in der Version 7.74.1 betrieben [89], mittlerweile hat es allerdings nicht mehr die gleiche Relevanz wie zu der Zeit von [110] und wird in dem aktuellen Report [80] nicht einmal mehr berücksichtigt. Aufgrund dieser Kriterien und dem Ausschluss der in [110] untersuchten Tools, wurden die drei oben benannten Tools ausgewählt. Sie erfüllen als einzige Tools aus [80] alle Kriterien. Zusätzlich ist es interessant genau diese drei Tools zu betrachten, da sie die gleiche Marktpräsenz haben [80].

Da in [80] keine Versionsnummern angegeben sind, wurde sich dafür entschieden die neusten Versionen der Tools zu betrachten. Bei Camunda wird allerdings die Version 7 [43] statt der Version 8 betrachtet, da die wichtigsten Features gleich geblieben sind und der größte Unterschied, der Web Modeler, nicht in der Open-Source- bzw. kostenlosen Variante von Camunda 8 zur Verfügung steht [44, 58]. Abgesehen von dem Web Modeler betreffen die meisten Veränderungen nur die technische Implementierung der zugrundeliegenden Engine [58] und einzelne Features der Version 8 befinden sich teilweise noch in der Entwicklung (s. digitaler Anhang - Öffentlich). Demnach wurde die Entscheidung getroffen für die Arbeit Camunda 7, als Variante, welche ausgereift und noch bis mindestens 2027 weiter supportet und entwickelt wird, zu betrachten [58]. Bei Axon Ivy und Bonitasoft werden die Versionen betrachtet, welche zu Beginn der Untersuchungen für die vorliegende Arbeit am aktuellsten waren [13, 39].

Alle drei Tools verfügen über zwei Bestandteile: dem Modeler und der Weboberfläche [14, 16, 27, 43]. Der sog. Modeler ist eine Desktop-Anwendung zum Modellieren des Prozesses bzw. Workflows und wird typischerweise zur Entwurfszeit verwendet [14, 16,

27, 43]. Die Weboberfläche wird genutzt, um die modellierten Prozesse bzw. Aspekte, welche zur Laufzeit relevant sind, auszuführen, zu administrieren und zu überwachen [14, 16, 27, 43]. Des Weiteren nutzen alle drei Tools den Begriff Task sowohl für die modellierte Aufgabe als auch für die Instanz der Task also das Workitem [18, 29, 52, 55].

### 2.5.1 Vorstellung und Bestandteile von Axon Ivy

Das erste Tool, welches analysiert wurde, ist der Axon Ivy Designer in der Version 11.2.1 [13]. Aus Gründen der Verständlichkeit wird das Tool im Folgenden als Axon Ivy bezeichnet. Axon Ivy ist ein hochmodernes Prozessmodellierungstool, durch welches die Geschäftsprozesse abgebildet werden können, sodass sie in der Weboberfläche durchlaufen werden können [3]. Ein vollständiger Workitem Lifecycle für Axon Ivy konnte nicht gefunden werden. Aufgrund der Liste sämtlicher Status, die Workitems in Axon Ivy annehmen können, lässt sich folgendes ableiten [18]: Ein Workitem befindet sich zuerst im Status „OPEN (SUSPENDED)“. Wenn eine Ressource das Workitem reserviert, wird es in den Status „OPEN (PARKED)“ versetzt und ist nun der Ressource zugeteilt. Sowohl aus dem Status „OPEN (SUSPENDED)“ als auch aus dem Status „OPEN (PARKED)“, kann ein Workitem gestartet werden und befindet sich anschließend im Status „IN_PROGRESS (RESUMED)“. Wenn ein Workitem erfolgreich abgeschlossen wurde, ändert sich der Status von „IN_PROGRESS (RESUMED)“ zu „DONE (DONE)“. Auch kann ein Workitem verzögert werden. Während der Verzögerung ist das Workitem im Status „DELAYED (DELAYED)“, bis die Verzögerung abgelaufen ist und das Workitem in den Status „OPEN (SUSPENDED)“ wechselt. Zusätzlich kann ein Workitem auch in sämtlichen bisher genannten Status bis auf den Status „DONE (DONE)“ abgebrochen bzw. zerstört werden, daraufhin wird der Status dann in „DESTROYED (DESTROYED)“ geändert. Dies beschreibt die für die vorliegende Arbeit relevanten Aspekte des Workitem Lifecycles. Für eine bessere Übersichtlichkeit wurde die Abbildung 4 aus den Informationen von [18] erstellt. Hierbei handelt es sich nur um einen Ausschnitt und nicht den kompletten Workitem Lifecycle.

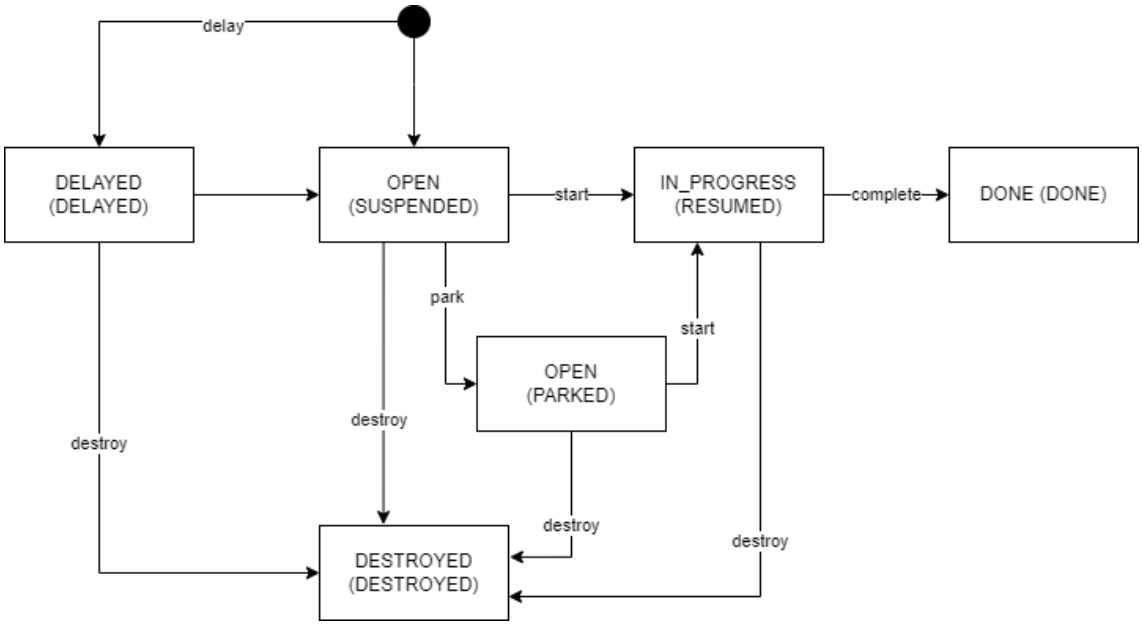


Abbildung 4: Ausschnitt aus dem Workitem Lifecycle von Axon Ivy nach [18]

Die sog. Developer Workflow UI ist die Standardmöglichkeit von Axon Ivy für Entwickler, um die Workflows zu testen und als Administrator Aufgaben zu erledigen [10]. Darüber hinaus verfügt Axon Ivy über einen sog. Market, auf dem zusätzliche Module kostenfrei

heruntergeladen werden können, um Axon Ivy zu erweitern [4]. Eines dieser Module ist das sog. Portal, welches vom Enduser genutzt wird, um Axon Ivy zu verwenden bzw. die Workitems zu bearbeiten [5]. Falls bei der Untersuchung der Patterns Unterschiede zwischen der Developer Workflow UI und dem Portal aufgefallen sind, wurde dies explizit in dem Kapitel „3.2 Durchführung der Untersuchung samt Bewertung der Patterns“ benannt. Im „Task“-Tab werden die Parameter für die Task und damit auch die Workitems der Task konfiguriert [8]. Hier kann unter anderem folgendes eingestellt werden [8]:

- Der Name der Task
- Der User bzw. die Rolle, dem bzw. der die Workitems der Task angeboten werden
- Die Priorität der Task
- Die Verzögerung der Task
- Das „Skip Tasklist“-Feld, ob die Workitems der Task sofort nach der Erstellung gestartet werden sollen
- Das Ablaufdatum oder der Ablaufzeitpunkt samt der neuen Ressourcenzuordnung, nachdem ein Workitem der Task abgelaufen ist
- Der individuelle Code, der in den Workitems der Task ausgeführt werden soll

Viele der Parameter in diesem Tab können per Dropdowns festgelegt werden [8]. In dem „Call“-Tab wird das aufzurufende Formular, bzw. hier User Dialog genannt, definiert und bestimmt, wie dieses gestartet werden soll [20]. Hier kann über den Button „Creates a new User Dialog“ ein neues Formular automatisch generiert werden [20]. Dafür muss in den anschließenden GUIs ausgewählt werden, welche Variablen im Formular belegt werden können [20]. Axon Ivy generiert das Formular anschließend automatisch und nutzt dieses für die Task [20]. Auch können Inputs für das Formular im „Mapping“-Fenster des „Call“-Tabs und auszuführender Code in einem „Code“-Fenster hinterlegt werden [20]. Im „Output“-Tab können die Outputs also die Daten, welche die Task ausgibt, konfiguriert werden [8]. Standardmäßig sind die Inputs der Task auch die Outputs, diese können aber manuell überschrieben werden [8]. Sowohl für die Festlegung der Outputs als auch für individuellen Code im „Code“-Fenster können die Funktions- und Attributs-Browser von Axon Ivy verwendet werden, um die passenden Werte zu hinterlegen [8]. Diese Browser können die Programmierung unterstützen, da sie sowohl eine Übersicht über vorhandene Methoden und Attribute liefern als auch den benötigten Code für die Umsetzung dieser Methoden und Attribute generieren und einfügen [7].

### 2.5.2 Vorstellung und Bestandteile von Bonitasoft

Das zweite betrachtete Tool ist die Bonita Studio Community Edition 2023.2 [28, 39]. Bonita Studio ist ein Teil von Bonita, der Open-Source- und erweiterbaren Plattform für die Automatisierung und Optimierung von Geschäftsprozessen [39]. Bonita Studio ist dabei das zentrale Tool um Prozesse, Datenmodelle und Anwendungen zu erstellen [28]. Bonita Studio beinhaltet eine Bonita Runtime, wodurch es geeignet ist, die erstellten Prozesse, Datenmodelle und Anwendungen für Testzwecke zu betreiben [28, 39]. Dies ist für die vorliegende Arbeit ausreichend und im Folgenden wird die Bonita Studio Community Edition 2023.2 aus Gründen der Verständlichkeit als Bonitasoft betitelt. Auch für Bonitasoft konnte kein Workitem Lifecycle identifiziert werden. Aufgrund der Auflistung sämtlicher möglichen Status lässt sich folgender Lifecycle ableiten [29]: Ein Workitem befindet sich während der Erstellung im Status „Initializing“. Anschließend wird das Workitem in den Status „Ready“ versetzt, bis es ausgeführt wird. Während der Ausführung wechselt das Workitem in den Status „Executing“. Aus diesem kann es entweder fehlschlagen und in den Status „Failed“ versetzt werden oder erfolgreich abgeschlossen werden, worauf der Status „Completed“ folgt. Auf diese Weise sind die relevanten Aspekte des Workitem Lifecycles für die vorliegende Arbeit dargestellt. Für eine bessere Übersichtlichkeit wurde die Abbildung 5 erstellt. Hierbei handelt es sich nur um einen Ausschnitt und nicht den kompletten Workitem Lifecycle.

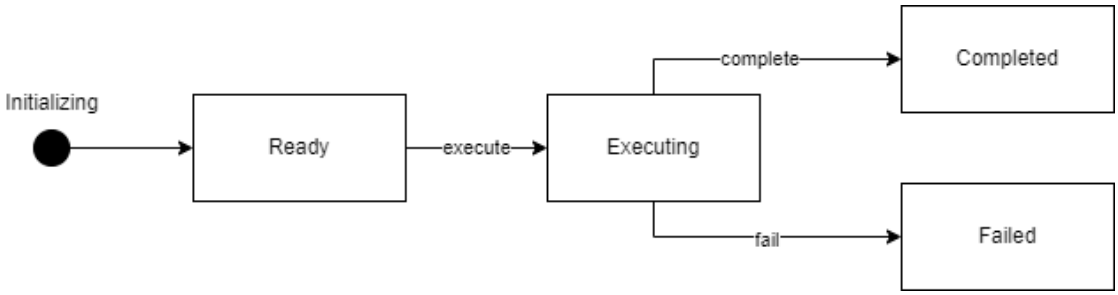


Abbildung 5: Ausschnitt aus dem Workitem Lifecycle von Bonitasoft nach [29]

Welche Ressource in Bonitasoft die Workitems einer Task ausführt, wird durch zwei Schritte festgelegt [34]. Zur Entwurfszeit muss ein sog. Actor für die Task festgelegt werden [34]. Ein Actor ist ein Platzhalter, der den potenziellen Benutzer darstellt, der den Schritt zur Laufzeit ausführen wird [34]. Bevor ein Prozess ausgeführt werden kann, müssen in einem zweiten Schritt jedem Actor tatsächliche User aus dem Organisationsmodell zugewiesen werden [34]. Die Actors dienen somit als Platzhalter und können angeben, welche Art von Usern eine Task übernehmen sollen, während die User echte Personen mit Username und Passwort sind, welche Workitems einer Task tatsächlich ausführen können [34]. Durch diese Zuweisung der User auf die Actors, das sog. Actor Mapping, wird festgelegt, welche konkreten User die Workitems der Tasks übernehmen [26, 34]. Um auf die GUI für das Actor Mapping zu gelangen, muss der „Configure“-Button geklickt werden [26]. In der GUI können dann jedem Actor beliebig viele Gruppen, Rollen, Memberships (eine Rolle in einer bestimmten Gruppe) oder spezifische User zugewiesen werden [26].

Im „General“-Tab können unter dem Unterpunkt „Actors“ die Actors ausgewählt werden, welche die Workitems der Tasks ausführen sollen [26]. Außerdem kann unter „Actors“ auch der sog. Actor Filter definiert werden [26]. Ein Actor Filter ist eine Möglichkeit die Benutzer variabel aus der Menge an potenziellen Usern herauszufiltern, die diese Task tatsächlich übernehmen können [25]. Auch in einer Swimlane existiert ein „General“-Tab mit dem Unterpunkt „Actors“, welcher über die gleichen Funktionen wie der bei Human Tasks verfügt [26]. Im „Data“-Tab können Variablen angelegt werden, sowohl in einer Human Task als auch in einer Swimlane bzw. dem Pool [35, 37]. In dem „Execution“-Tab einer Human Task kann ein sog. Contract definiert werden [30]. Unter dem Punkt „Contract“ werden sowohl die Inputs festgelegt, die die Task benötigt, um zu starten, als auch Constraints, welche prüfen ob die Werte der Inputs valide sind [30]. Basierend auf diesen Contracts kann automatisch ein Formular im „Execution“-Tab unter „Form“ erstellt werden [31, 38]. Hierfür wird die Entwicklungsumgebung von Bonitasoft für die Erstellung von Formularen, der sog. UI Designer, verwendet, welcher als Formulartyp ausgewählt werden kann [38]. Auch können im „Execution“-Tab unter „Operations“ sog. Operations definiert werden [33]. Eine Operation aktualisiert den Wert einer für die Task oder auf Prozessebene definierten Variablen, während das Workitem der Task ausgeführt wird [33].

### 2.5.3 Vorstellung und Bestandteile von Camunda

Das dritte Tool, welches analysiert wurde, ist Camunda 7 Community in der Version 7.21 [43, 47]. Hierbei handelt es sich um eine light-weight Open-Source-Plattform für Business Process Management [43]. Aus Gründen der Verständlichkeit wird das Tool im Folgenden als Camunda bezeichnet. Camunda beschreibt den eigenen Workitem Lifecycle in Gänze in der Abbildung 6. Aus [52] können sämtliche Zustandstransfers und Status der Workitems abgelesen werden.

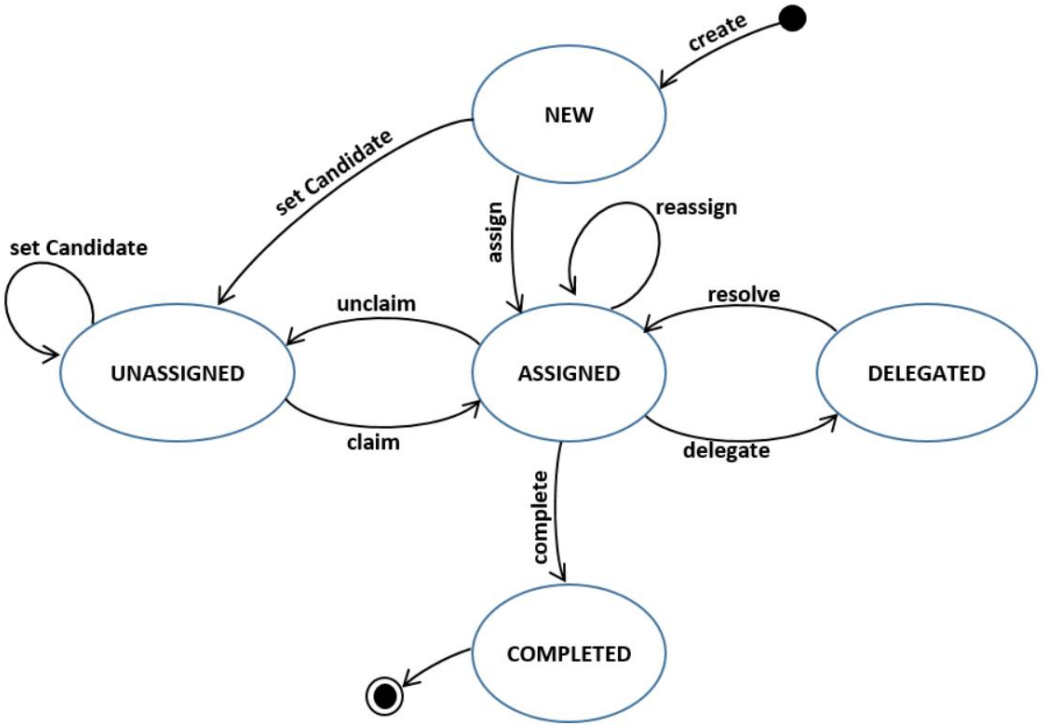


Abbildung 6: Workitem Lifecycle von Camunda [52]

Die Weboberfläche von Camunda ist in drei Teile aufgeteilt: „Admin“, „Cockpit“ und „Tasklist“ [42, 45, 53]. „Tasklist“ wird genutzt, um an User Tasks zu arbeiten [53]. „Admin“ ist eine Anwendung, mit der die Benutzer und Gruppen über den Identity Service der Engine und Berechtigungen über den Authorization Service der Engine konfiguriert werden können [42]. „Cockpit“ ist die Webanwendung für die Überwachung und den Betrieb der modellierten Prozesse [45]. Allerdings können in der Community Version nur laufende Prozessinstanzen und Task betrachtet werden, wobei auch bei diesen der Status nicht einsehbar ist [48]. Im Modeler können Skripte einfach in viele verschiedene BPMN-Elemente eingefügt bzw. hinterlegt werden [51]. Im „Forms“-Tab einer User Task können auch Formulare erstellt werden, sowohl manuell als auch bei den sog. Generated Task Forms automatisch [54]. In einer User Task kann außerdem im „Task listeners“-Tab ein sog Task Listener definiert werden [46]. Ein Task-Listener wird verwendet, um individuellen Code beim Auftreten eines, auf das Workitem der User Task bezogenen, Ereignisses auszuführen [46]. Zu diesen Ereignissen gehören, die Erstellung, das Updaten, die Zuteilung, das Ablaufen, das Fertigstellen oder das Löschen des Workitems [46].

### 2.5.4 Vorstellung von YAWL als zusätzliches Beispieltool

YAWL (Yet Another Workflow Language) wurde von den Professoren van der Aalst und ter Hofstede, welche die Workflow Patterns Initiative leiten, entwickelt [93, 105]. YAWL betitelt sowohl die entwickelte Modellierungssprache als auch das dazugehörige Open-Source-WfMS, welches YAWL implementiert [93, 105, 107]. Für die vorliegende Arbeit wurde YAWL 5.0 betrachtet [104], welches aus Gründen der Verständlichkeit als YAWL betitelt wird. YAWL wurde sowohl auf Grundlage der Workflow Patterns als auch auf Grundlage von Petri-Netzen entwickelt, um zu zeigen, dass eine umfassende Unterstützung der Workflow Patterns durch eine Modellierungssprache bzw. ein WfMS möglich ist [93, 105, 107]. Dementsprechend sollen auch Resource Patterns durch YAWL umfassend abgebildet werden können [93].

In der vorliegenden Arbeit wird YAWL außer Konkurrenz zu den restlichen Tools untersucht, da YAWL kein BPMN verwendet. Wie bereits erwähnt wurde YAWL mit dem Ziel entwickelt, die Workflow Patterns abbilden bzw. umsetzen zu können [93, 105, 107].

Dementsprechend wird YAWL als ein zusätzliches Beispieltool im Vergleich zu den restlichen Tools betrachtet und dient so dem Zweck zu vergleichen bzw. darzustellen, wie bestimmte Patterns umgesetzt werden könnten und welche Patterns selbst in YAWL zu Problemen führen. Das Ziel der vorliegenden Arbeit ist allerdings die Umsetzung von Resource Patterns in BPMN basierten WfMS zu untersuchen und basierend darauf eine Tool-Empfehlung abgeben zu können. Daher werden in dem Kapitel „3.2 Durchführung der Untersuchung samt Bewertung der Patterns“ nur die Untersuchungen zu den drei BPMN basierten Tools detailliert erläutert. Die Endergebnisse für YAWL werden in den Abbildungen 7 bis 14 hinterlegt und in dem Kapitel „3.3.1 Vergleich der Tools aufgrund der Untersuchungsergebnisse der Patterns“ kurz beschrieben. Die detaillierten Beschreibungen für YAWL sind in der Datei „Detaillierte_Bewertungsübersicht_der_Patterns.xlsx“ im digitalen Anhang zu finden. Da YAWL außer Konkurrenz betrachtet wird, wird hier auch nicht auf den Workitem Lifecycle von YAWL eingegangen. Die Abbildungen über die verschiedenen Zustandstransfers und Status der Workitems in YAWL können im digitalen Anhang in den Dateien „YAWL_Workitem Lifecycle.PNG“ [71] und „YAWL_Workitem State Chart.pdf“ [103] betrachtet werden.

# 3 Untersuchung der Resource Patterns in den ausgewählten Tools

## 3.1 Erläuterung des Bewertungsschemas für die Untersuchung

Für eine einheitliche und systematische Bewertung der einzelnen Patterns in den Tools wurde ein entsprechendes Bewertungsschema erarbeitet, welches im Folgenden vorgestellt wird. Dabei wurde sich am Vorgehen von [84] orientiert. In dem Kapitel „F Evaluation Criteria“ aus [84] werden pro Pattern Bewertungskriterien bestimmt und je nachdem welche Kriterien erfüllt werden, erhält das Pattern eine Bewertung von „+“, „+/-“ oder „-“. Dieses Schema wurde als nicht geeignet für die vorliegende Arbeit gewertet, da die Bewertungen durch die drei Möglichkeiten sehr begrenzt sind und wenig differenziert sein können. Dementsprechend wurde ein neues Schema erarbeitet, welches den Ansatz von einzelnen Bewertungskriterien pro Pattern weiterverwendet.

Das in der vorliegenden Arbeit angewandte Bewertungsschema besteht aus vier Teilen: den Bewertungskriterien, der Gewichtung, den Punktzahlen nach einer Skala und dem Endergebnis.

Für die Bewertungskriterien wurden die wichtigsten Merkmale eines Patterns aus den Abschnitten mit den Überschriften „Description“, „Overview“, „Implementation“ und ggf. „Issues“ von [93] herausgearbeitet. Somit stammen alle Kriterien aus [93], wobei das erste Bewertungskriterium immer die Beschreibung des Patterns darstellt. Dies wurde für jedes Pattern durchgeführt. Anschließend wurde für jedes Bewertungskriterium eine Gewichtung festgelegt. Hierbei konnte zwischen zwei Gewichtungsmöglichkeiten gewählt werden: Die Gewichtung „*1“ für Kriterien, welche die grundlegende Funktionalität des Patterns abbilden und die Gewichtung „*0,5“ für technische Details oder zusätzliche weiterführende Anforderungen. Falls Kriterien aus dem Abschnitt „Issues“ herangezogen wurden, wurden diese als zusätzliche weiterführende Anforderungen eingestuft. Um diese gewichteten Bewertungskriterien systematisch zu beurteilen, wurden Punktzahlen nach einer Skala vergeben. Diese Skala hat drei Kategorien:

- „Standard“, für eine native Umsetzung des Kriteriums im Tool, ohne dass hierfür Code geschrieben werden muss.
- „Teilweise programmiert“, für eine Umsetzung des Kriteriums, bei der nur die standardmäßigen Funktionen des Tools nicht ausreichen und mindestens eine Zeile Code benötigt wird. Wichtig ist es hier zu unterschieden, dass der Code nicht genutzt wird, um die gesamte Logik des Patterns umzusetzen. Die Logik wird weiterhin durch die Standardmöglichkeiten des Tools abgebildet und der Code dient als Unterstützung durch den z.B. Daten herangezogen werden oder Ähnliches.
- „Logik programmiert“, für Kriterien, welche nur umgesetzt werden können, wenn die gesamte Logik des Patterns programmiert wurde. Jegliche Kriterien, bei denen die grundlegende Logik des Patterns nicht durch das Tool umgesetzt werden kann, sondern komplett eigenständig programmiert werden muss, fallen in diese Kategorie.

Innerhalb der ersten beiden Kategorie gibt es zwei verschiedene Punktzahlen, um Unterschiede bei der Umsetzung zwischen den Tools besser in der finalen Punktzahl abbilden zu können. In der vorliegenden Arbeit sollen die Funktionalitäten in den Tools für die Umsetzung der Resource Patterns untersucht werden. Hierbei soll nicht betrachtet werden wie gut eine eigenständig programmiert Umsetzung der Patterns in einem Tool angebunden werden kann. Daher erhalten Tools für ein Kriterium in der Kategorie „Logik programmiert“ 0 Punkte und entsprechend in der Kategorie „Standard“ am meisten Punkte:

- „Standard“: Ein Tool erhält in einem Bewertungskriterium in dieser Kategorie 4 Punkte, wenn sich das Kriterium im Tool perfekt umsetzen lässt, ohne dass Probleme auftreten. Falls es bei der Umsetzung Probleme gibt oder sich das Kriterium

in einem anderen Tool noch besser implementieren lässt, werden 3 Punkte vergeben.

- „Teilweise programmiert“: Ein Tool erhält für ein Bewertungskriterium in dieser Kategorie 2 Punkte, wenn für die Umsetzung mindestens eine Zeile Code benötigt wird, diese aber keine Logik enthält, sondern nur zur Unterstützung dient, um z.B. Daten zu erhalten. Falls es bei der Umsetzung Probleme gibt oder sich das Kriterium in einem anderen Tool noch besser implementieren lässt, erhält das Tool für das Kriterium nur 1 Punkt.
- „Logik programmiert“: Ein Tool erhält für ein Bewertungskriterium in dieser Kategorie immer 0 Punkte. Ein komplett eigenständig programmiertes Modul samt Logik und ggf. GUI-Elementen kann nicht als Funktionalität des Tools betrachtet werden, nur da das Tool eine Schnittstelle für benutzerdefinierten Code vorweist. Daher werden in einem solchen Fall 0 Punkte vergeben, unabhängig davon, ob das Tool vorgefertigte Mittel wie z.B. ein Application Programming Interface (API) zur Verfügung stellt, welche die Umsetzung der Logik unterstützen. Da eine Implementierung der Logik durch benutzerdefinierten Code nicht im Fokus der Arbeit liegt, wird diese nicht weiter betrachtet bzw. umgesetzt. Daher kann nicht eindeutig festgelegt werden, ob ein Kriterium in einem Tool gar nicht umsetzbar oder nur durch programmierte Logik umsetzbar ist. Deswegen wird in der vorliegenden Arbeit keine Unterscheidung gemacht und beides wird in diese Kategorie eingeordnet und entsprechend mit 0 Punkten bewertet.

Für die Einordnung eines jeden Kriteriums in die passende Kategorie, wird versucht das jeweilige Kriterium in jedem Tool umzusetzen. Die Umsetzungen werden in dem Kapitel „3.2 Durchführung der Untersuchung samt Bewertung der Patterns“ beschrieben. Sollte es möglich sein, das Kriterium erfolgreich abzubilden, kann aufgrund der Ergebnisse die passende Kategorie bzw. Punktzahl aus „Standard“ und „Teilweise programmiert“ gewählt werden. Die Datei mit der erfolgreichen Implementierung des Kriteriums in dem jeweiligen Tool begründet somit die Einordnung. Diese Dateien können im digitalen Anhang eingesehen werden. Falls das Kriterium nicht umgesetzt werden kann, wird es in die Kategorie „Logik programmiert“ eingeordnet. Da in einem solchen Fall keine Implementierung vorhanden ist, um die Einordung zu validieren, wird online nach passenden Quellen gesucht. Sollte keine Quelle gefunden werden, die bestätigt, dass das Kriterium gar nicht oder nur durch individuell programmierte Logik umsetzbar ist, wird in dem jeweiligen Forum des Tools ein Beitrag veröffentlicht. In diesem wird nachgefragt, wie das Kriterium bzw. die entsprechende Funktionalität in dem Tool umgesetzt werden kann. Aufgrund der begrenzten Bearbeitungsdauer der vorliegenden Arbeit werden nur Antworten berücksichtigt, welche auf die jeweiligen Forenbeiträge innerhalb eines Monats nach Veröffentlichung erfolgen. Sollte ein Forenbeitrag nach einem Monat keine Antworten haben, wird dies als Bestätigung gewertet, dass das Kriterium nicht umsetzbar ist. Auf diese Weise wird die Einordnung des Kriteriums validiert. Die entsprechenden Forenbeiträge sind im digitalen Anhang zu finden.

Für jedes Pattern wird pro Bewertungskriterium die Punktzahl mit der Gewichtung multipliziert. Die Ergebnisse werden für das Pattern aufsummiert und auf eine Punktzahl von 10 normiert. Dabei wird auf die zweite Nachkommastelle gerundet. Auf diese Weise erhält jedes Pattern als Endergebnis pro Tool eine Punktzahl auf einer Skala von 0 bis 10. Die Anwendung des oben beschriebenen Schemas auf die Patterns kann in der Datei „Detaillierte_Bewertungsübersicht_der_Patterns.xlsx“ eingesehen werden.

## 3.2 Durchführung der Untersuchung samt Bewertung der Patterns

### 3.2.1 Untersuchung der Creation Patterns

Wie im Bewertungsschema beschrieben, ergeben sich für das Pattern „Direct Distribution“ die folgenden zwei Kriterien mit den entsprechenden Gewichtungen:

- 1. Die Möglichkeit, zur Entwurfszeit die Identität der Ressource(n) anzugeben, der zur Laufzeit die Workitems dieser Task zugeordnet werden
    - Gewichtung: *1
- 2. Die Möglichkeit mehrere Ressourcen einzeln anzugeben
    - Gewichtung: *0,5

Axon Ivy erhält für das 1. Kriterium 3 Punkte, da der Name eines Users in der gewünschten Task nur als String hinterlegt werden kann. Auf diese Weise kann es potenziell zu Rechtschreibfehlern kommen und die Usernamen von jedem User müssen bei der Erstellung vorliegen. Dies wird von Bonitasoft besser umgesetzt. Grundsätzlich erfüllt Axon Ivy das Kriterium aber nativ, indem in einer User Task unter dem Tab „Task“ bei „Responsible“ das Dropdownfeld „User from Attr." ausgewählt und dort der Username, also die ID des Users, als String in Anführungszeichen hinterlegt wird. Die Workitems dieser Task werden anschließend dem eingetragenen User angeboten und automatisch nur noch diesem angezeigt. Eine andere Möglichkeit wäre es, eine Rolle zu erstellen welche nur einen User beinhaltet. Dann könnte unter „Responsible“ das Feld „Role" ausgewählt werden und die gewünschte Rolle aus einem Dropdown-Menü gewählt werden. Zur Entwurfszeit können beliebig viele User angelegt werden. Dafür liegt in dem im Tool angezeigten „Config“-Ordner die „user.xml“-Datei vor. Die Erstellung der User erfolgt über eine GUI.
In dem 2. Kriterium erzielt Axon Ivy 0 Punkte, da es selbst mithilfe von individuell programmierter Logik nicht möglich ist, das Kriterium umzusetzen, ohne das Konzept der Rollen zu verwenden (s. digitaler Anhang - Beantwortet). Das Konzept von Rollen würde es ermöglichen, ein Workitem mehreren Usern zuzuordnen. Beim zweiten Kriterium soll es aber möglich sein, die User einzeln anzugeben, sodass nicht für jede Konstellation an Usern eine neue Rolle angelegt werden muss. In dem „User from Attr."-Feld kann aber immer nur ein Username gleichzeitig hinterlegt werden. Es gibt keine Möglichkeit, mehrere Ressourcen einzeln anzugeben (s. digitaler Anhang - Beantwortet). Die einzige Möglichkeit für das gewünschte Verhalten beinhaltet die Verwendung von Rollen samt zusätzlich programmierter Logik (s. digitaler Anhang - Beantwortet).
Zusammengerechnet erhält Axon Ivy für das Pattern „Direct Distribution“ somit die Gesamtpunktzahl 5 von 10. Zusätzlich ist anzumerken, dass Axon Ivy im Kontrast zu BPMN [93], nicht die Möglichkeit bietet, Pools zu verwenden, um Usern die Workitems einer Task zuzuordnenden. Die Pools bzw. Swimlanes in Axon Ivy dienen nur zur Visualisierung und haben keinen Einfluss während der Ausführung [17].

Bonitasoft erhält für das 1. Kriterium 4 Punkte, da standardmäßig zur Entwurfszeit die User für das Actor Mapping aus einer Liste ausgewählt werden können und nicht als String eingetragen werden müssen. Somit gibt es sowohl kein Risiko für Rechtschreibfehler als auch eine vordefinierte Liste aller Namen bei der Erstellung, sodass die gewünschten User nur ausgewählt werden müssen. In einem Pool bzw. einer Swimlane kann ein Actor hinterlegt werden, welchem später konkrete User zugewiesen werden. Im „General“-Tab der Swimlane können unter „Actors“ beliebige viele Actors angelegt werden. In jeder Human Task kann im „General“-Tab unter „Actors“ ausgewählt werden, ob der Actor der Swimlane verwendet werden soll oder ein Actor separat in der Task angegeben werden soll. Diese agieren als Platzhalter für Ressourcen während der Modellierung und vor der Ausführung des Prozesses erfolgt eine Zuweisung der Actors auf die echten User aus dem Organisationsmodell. In der Datei „Organization.xml“ kann das Organisationsmodell mithilfe von entsprechenden GUIs für Gruppen, Rollen und User erstellt werden. Hier können zur Entwurfszeit beliebig viele User angelegt werden. Nach dem Modellieren kann auf den Button „Configure“ geklickt werden, um in der GUI unter dem Punkt „Actor mapping“ jedem Actor entsprechende Entitäten aus dem Organisationsmodell zuzuweisen. Dabei ist es möglich, sowohl einen als auch mehrere echte User auf einen Actor zuzuweisen. Anschließend muss ein echter User unter „Authentication“ angegeben werden und das Modell kann bereitgestellt werden (deployed). Zur Laufzeit sind die Workitems einer Task dann für jeden User sichtbar, der dem Actor dieser Task zugewiesen wurde.

Auch für das 2. Kriterium erhält Bonitasoft 4 Punkte, da durch das Actor Mapping die Zuweisung von beliebig vielen Usern auf einen Actor möglich ist und diese User somit einzeln angegeben werden können.
Somit erzielt Bonitasoft für das Pattern „Direct Distribution“ die Gesamtpunktzahl 10 von 10. Beide Möglichkeiten der Userzuweisung aus BPMN [93], sowohl durch die Verwendung von Pools als auch durch die direkte Zuordnung in einer Task, sind in Bonitasoft abbildbar.

Das 1. Kriterium wird in Camunda mit 3 Punkten bewertet, da der Username in der Task als String eingetragen werden muss und somit die gleichen Probleme wie bei Axon Ivy vorliegen. Trotzdem kann das Kriterium nativ in Camunda umgesetzt werden, indem zuerst die User in der „Admin“-Weboberfläche angelegt werden. Anschließend können die Namen der User zur Entwurfszeit im Modeler verwendet werden. In einer User Task können die Usernamen, also die IDs der User, als String in dem Feld „Assignee“ oder „Candidate Users“ hinterlegt werden. Die Workitems der entsprechenden Task werden nur den eingetragenen Usern angezeigt. Ein Workitem wird einem User, der im Feld „Assignee“ eingetragen ist, fest zugeteilt, während es einem User der im Feld „Cadidate Users“ steht, nur angeboten wird. Zusätzlich zu den Usern müssen auch passende Berechtigungen in der „Admin“-Weboberfläche unter „Authorizations“ angelegt werden, damit die User überhaupt auf die „Tasklist“-Weboberfläche zugreifen dürfen.
Camunda erhält für das 2. Kriterium 3 Punkte, da es zwar standardmäßig möglich ist, im Feld „Cadidate Users“ der User Task mehrere User einzeln anzugeben, aber die Problematik der Namenseingabe als String weiterhin besteht. Ein Workitem wird dem eingetragenen User im Feld „Assignee“ der User Task direkt fest zugeteilt, weshalb hier immer nur ein User hinterlegt werden kann. Im Feld „Cadidate Users“ können auch mehrere User als String hinterlegt werden, wenn diese durch ein Komma getrennt sind. Hier werden die Workitems der Task dann sämtlichen eingetragenen Usern angeboten.
Camunda erzielt somit für das Pattern „Direct Distribution“ die Punktzahl 7,5 von 10. Auch in Camunda können, im Gegensatz zu BPMN [93], keine Pools bzw. Swimlanes verwendet werden, um den Workitems einer Task User zuzuordnen.

Für das nächste Pattern „Role-Based Distribution“ wurden die folgenden fünf Bewertungskriterien untersucht:

- 1.1 Die Möglichkeit, zur Entwurfszeit eine Rolle anzugeben, der zur Laufzeit die Workitems dieser Task zugeordnet werden
    - Gewichtung: *1
- 1.2 Die Möglichkeit, zur Entwurfszeit mehrere Rollen anzugeben, denen zur Laufzeit die Workitems dieser Task zugeordnet werden
    - Gewichtung: *1
- 1.3 Rollen sind vorhanden und dienen als Mittel zur Gruppierung von Ressourcen mit ähnlichen Merkmalen
    - Gewichtung: *1
- 1.4 Wenn ein Workitem auf diese Weise zugeordnet wird, wird es allen Ressourcen zugeordnet, die Mitglieder der mit der Task verbundenen Rolle(n) sind
    - Gewichtung: *1
- 2. Beide Konzepte, Rollen und Gruppen, existieren und sind getrennt voneinander
    - Gewichtung: *0,5

Sowohl das Kriterium 1.1 als auch das 1.3 ist nativ in Axon Ivy umsetzbar, wodurch das Tool jeweils 4 Punkte in den Kriterien erzielt. In der „roles.xml“-Datei von Axon Ivy können zur Entwurfszeit Rollen erstellt werden. Jedem User können, in der GUI für die Erstellung bzw. Bearbeitung der User, beliebig viele Rollen zugewiesen werden. In einer User Task kann danach unter „Responsible“ das Feld „Role“ angegeben werden, um aus einem Dropdown die gewünschte Rolle für die Task auszuwählen. Sämtlichen Usern, welche über die gewählte Rolle verfügen, werden zur Laufzeit die Workitems der Task zugeordnet.

Für das Kriterium 1.2 erhält Axon Ivy 3 Punkte, da es durch die verschiedenen Rollen-Konzepte zwar möglich ist, standardmäßig die Workitems einer Task mehreren Rollen zuzuordnen, aber immer nur eine Rolle direkt in der Task angegeben werden kann. Dieses Problem kann immer nur durch eine Verwendung der zwei Rollen-Konzepte umgangen werden. In einer Task lässt sich im Dropdown-Menü für „Responsible“ immer nur eine Rolle gleichzeitig auswählen. Um trotzdem mehrere Rollen zu hinterlegen, denen zur Laufzeit die Workitems zugeordnet werden sollen, müssen die Rollen in der „roles.xml“-Datei miteinander verknüpft sein. Axon Ivy beinhaltet zwei Rollen-Konzepte, Permission Roles und Member Roles. Jede neu angelegt Rolle ist erstmal eine Member Role. Sollen nun mehrere Rollen in einer Task angegeben werden, können die Rollen unter einer bestehenden oder neu angelegten Rolle als Permission Role hinterlegt werden, indem sie durch den „add“-Button in der Rollen-GUI hinzugefügt werden. Somit kann die Member Role in der Task angegeben werden und alle Permission Roles dieser Member Role werden auch die Workitems der Task sehen können.
Axon Ivy erhält bei dem Kriterium 1.4 volle 4 Punkte, da zur Laufzeit die Workitems einer Task immer nur den Usern angezeigt werden, welche die in der Task hinterlegte Rolle oder eine entsprechende Unterrolle dieser Rolle besitzen. In der „roles.xml“-Datei können durch den „new“-Button und den „add“-Button Hierarchien aus den Rollen erstellt werden. Es können nicht nur Permission Roles als Unterrollen einer Member Role hinzugefügt werden. Auch Member Roles können, durch den „new“-Button in der Rollen-GUI, als Unterrolle einer anderen Member Role angelegt werden.
Axon Ivy erhält für das 2. Kriterium 3 Punkte, da es durch die zwei Rollen-Arten zwar verschiedene Konzepte bietet, welche die Funktionalitäten von Rollen und Gruppen umsetzen können, diese aber nicht getrennt voneinander sind. Trotz der beiden Rollen-Konzepte bietet Axon Ivy keine Möglichkeit, Gruppen anzulegen [15]. Es existiert nur das Konzept der Rollen. Allerding ist es möglich, mithilfe von Member Roles eine Organisationsstruktur abzubilden und durch Permission Roles Berechtigungskonzepte zu erstellen. Dies bildet die Funktionalitäten von Gruppen und Rollen größtenteils ab.
Abschließend ergibt sich bei Axon Ivy für das Pattern „Role-Based Distribution“ eine Punktzahl von 9,17 von 10.

In Bonitasoft sind die Kriterien 1.1 bis 1.4 standardmäßig durch das Actor Mapping möglich und erhalten jeweils 4 Punkte. In der Organisationsmodell-Datei „Organization.xml“ können Rollen und Gruppen erstellt werden. Diese können einem User in der entsprechenden User-GUI zugewiesen werden. Wichtig ist hierbei anzumerken, dass einem User eine Rolle nur in Verbindung mit einer Gruppe zugewiesen werden kann. Ein User kann beliebig viele Rollen erhalten. Wie beim „Direct Distribution“-Pattern beschrieben, erfolgt anschließend zur Entwurfszeit das Actor Mapping, wobei einem Actor beliebig viele Rollen zugewiesen werden können. Die Workitems der Task werden allen Usern angeboten, die mindestens eine der Rollen besitzen, welche auf den Actor der Task zugewiesen wurden.
Für das 2. Kriterium erhält Bonitasoft 4 Punkte, da es standardmäßig sowohl Gruppen als auch Rollen gibt und diese getrennt voneinander existieren, aber auch zusammen genutzt werden können. Wie bereits erwähnt können zusätzlich zu Rollen auch Gruppen angelegt werden. Bei der Zuweisung zu einem User müssen die Konzepte immer in Kombination genutzt werden, sodass ein User nur eine Rolle erhalten kann, wenn auch eine Gruppe, zu der die Rolle gehört, mit angegeben wird. Diese Kombination wird in Bonitasoft Membership genannt. Gruppe und Rollen sind getrennt voneinander und können im Actor Mapping sowohl separat als auch in Kombination durch die Memberships verwendet werden. Eine detailliertere Beschreibung des Actor Mappings folgt im Pattern „Organisational Distribution“.
Zusammengerechnet erzielt Bonitasoft für das Pattern „Role-Based Distribution“ die Punktzahl 10 von 10.

Camunda erhält für das 2. Kriterium 0 Punkte, da es nur über das Konzept von Gruppen verfügt (s. digitaler Anhang – Öffentlich) [49] . Diese können in der „Admin“-Weboberfläche erstellt werden. Anschließend kann ein User, in den User-Einstellungen der „Admin“-

Weboberfläche, beliebig vielen Gruppen hinzugefügt werden. Grundsätzlich werden die Gruppen eher wie Rollen verwendet und verfügen über keine Möglichkeit, miteinander verknüpft oder in einer Hierarchie angeordnet zu werden (s. digitaler Anhang – Beantwortet). Demnach wird das Gruppen-Konzept hier wie das Rollen-Konzept untersucht und bewertet. Trotzdem ist nur das Gruppen-Konzept vorhanden, welches nicht wie bei Axon Ivy über verschiedene Arten oder Möglichkeiten verfügt, die Funktionalitäten von Rollen und Gruppen abzudecken (s. digitaler Anhang – Beantwortet).
Die Kriterien 1.1, 1.2 und 1.4 sind nativ erfüllt. Die angelegten Gruppen können im Modeler verwendet werden. Dort können die Gruppennamen, also die IDs der Gruppen, bei User Tasks im Feld „Candidate Groups“ als String eingefügt werden. Hier können beliebig viele Gruppen hinterlegt werden, indem die Namen durch Kommas getrennt werden. Zur Laufzeit werden die Workitems einer User Task allen Usern angeboten, welche Teil von mindestens einer der im „Candidate Groups“-Feld hinterlegten Gruppen sind. Das Kriterium 1.4 erhält somit 4 Punkte. Die anderen beiden Kriterien erhalten 3 Punkte, da die Gruppennamen nur als String hinterlegt werden können. Dadurch ist Potenzial für Rechtschreibfehler vorhanden und bei der Erstellung müssen sämtliche Gruppennamen bekannt sein. Es ist wichtig anzumerken, dass für jede neu erstellte Gruppe in „Authorizations“ zuerst eine Berechtigung für die Gruppe angelegt werden muss, sodass diese auf „Tasklist“ zugreifen darf.
Camunda erzielt 3 Punkte im Kriterium 1.3, da Gruppen zwar nativ verwendet werden können, um User zu gruppieren, allerdings verfügen sie über keine Möglichkeit, diese in Beziehung zueinander zu stellen bzw. Hierarchien aufzubauen (s. digitaler Anhang – Beantwortet). Somit sind verschiedene Gruppierungen von Ressourcen nicht oder nur sehr umständlich abzubilden.
Final erhält Camunda somit für das Pattern „Role-Based Distribution“ die Punktzahl 7,22 von 10.

Bei dem Pattern „Deferred Distribution“ wurden die folgenden drei Bewertungskriterien betrachtet:

- 1. Die Möglichkeit, zur Entwurfszeit festzulegen, dass die Identifizierung der Ressource(n), der die Workitems einer Task zugeordnet werden, bis zur Laufzeit aufgeschoben wird
    - Gewichtung: *1
- 2. Es kann ein Zuordnungsmechanismus (Business Rule, Data Elements, ...) deklariert werden, der zur Laufzeit evaluiert wird und so die Workitems einer oder mehreren Ressource(n) oder Rolle(n) zuordnet
    - Gewichtung: *1
- 3. Es kann deklariert werden, was passieren soll, wenn der Zuordnungsmechanismus keine passende Ressource identifiziert
    - Gewichtung: *0,5

Axon Ivy erzielt im 1. Kriterium 4 Punkte, da in einer User Task nativ sowohl eine Variable verwendet werden als auch keine Angabe gemacht werden kann, um die Identifizierung der Ressource auf die Laufzeit zu verschieben. Um eine Variabel anzulegen, muss zuerst in dem „data clases“-Ordner eine Datei, z.B. „Data“, erstellt werden, welche eine Klasse für Daten darstellt. In der angelegten Datei „Data“ kann anschließend eine String-Variable als Attribut der Klasse angelegt werden. Ein Objekt dieser Klasse wird Data Element genannt. In dem „Responsible“-Feld der User Task kann dann die Option „User from Attr.“ ausgewählt und die Variable mit dem Präfix „in.“ hinterlegt werden. Zur Laufzeit kann die Variable entweder durch ein Skript oder ein Formular belegt werden. Ein Formular bzw. hier User Dialog genannt, kann per GUI im „Call“-Tab der User Task automatisch angelegt werden. Über GUIs kann angegeben werden, welche Variablen im Formular enthalten sein sollen. Anschließend erstellt Axon Ivy das Formular automatisch. Eine andere Möglichkeit wäre das Feld „User from Attr.“ freizulassen. Auf diese Weise wird das Workitem nur dem Administrator angezeigt und dieser kann zur Laufzeit mithilfe der „Delegate“-Funktion entscheiden, wer das Workitem erhalten soll. Die „Delegate“-Funktion wird im Pattern „Delegation“ näher erklärt.

In dem 2. Kriterium erhält Axon Ivy 4 Punkte, da Data Elements standardmäßig für den Zuordnungsmechanismus verwendet werden können. Mit den Optionen „User from Attr." und „Role from Attr." ist es möglich, mithilfe einer Variable Workitems sowohl einem User als auch einer Rolle anzubieten. Sämtliche Konfiguration der Variable und auch das Erstellen der Variable, der Formulare oder ggf. des Skripts, erfolgt zur Entwurfszeit und die Auswertung bzw. Zuordnung zur Laufzeit. Ein Skript wäre eine Lösung, welche die Logik des Kriteriums programmiert. Da es aber die Möglichkeit gibt, die Logik des Kriteriums nativ mithilfe von Formularen umzusetzen, wird diese Option für die Bewertung betrachtet.
Für das 3. Kriterium erhält Axon Ivy 0 Punkte, da es nur umgesetzt werden kann, indem die Logik programmiert wird. Es liegt in Axon Ivy keine Möglichkeit vor, die Outputs des Formulars bzw. die entsprechenden Inhalte der Variablen zu prüfen. Dementsprechend lässt sich dies nur durch Code umsetzen.
Abschließend erzielt Axon Ivy somit bei dem Pattern „Deferred Distribution" die Punktzahl 8 von 10.

Bonitasoft erhält für das 1. Kriterium 2 Punkte, da ein passender Actor Filter zwar nativ verfügbar ist, aber trotzdem Code benötigt wird, um das gewünschte Verhalten abzubilden. Der Actor Filter „Find single User" wird verwendet, indem dieser im „General"-Tab einer Human Task oder Swimlane unter „Actors" ausgewählt wird. Dieser Actor Filter teilt mithilfe einer User-ID ein Workitem diesem User zu. Allerdings ist die hier gewünschte User-ID nicht der Name des Users. Daher muss in dem Input für den Actor Filter Code verwendet werden, um einen User über seinen Namen zu suchen und anschließend dessen ID auszugeben. Die Suchfunktion kann Bonitasoft aus vorgefertigten Befehlen aus dem Browser per Klick generieren und auch die Funktion, um die ID des Users zu erhalten ist verfügbar. Als Input für den Actor Filter muss eine Variable genutzt werden, auf welcher der gewünschte Username gespeichert wird. Diese Variable kann mithilfe eines Formulars oder eines Skripts einen Wert erhalten. Zusätzlich muss der Actor Filter in einer Human Task oder Swimlane eingesetzt werden, um die Identifizierung der Ressource dynamisch zur Laufzeit durchzuführen (s. digitaler Anhang – Öffentlich). Der Actor Filter überschreibt das Actor Mapping für die Task oder Swimlane. Ohne einen Actor Filter muss das Actor Mapping verwendet werden, bei diesem müssen zur Entwurfszeit schon fest User oder Rollen angegeben werden und es können keine Inputs wie Variablen verwendet werden (s. digitaler Anhang – Öffentlich). Eine Festlegung zur Laufzeit ist dementsprechend durch das Actor Mapping nicht möglich. Bonitasoft verfügt über vorgegebene Actor Filter, die Auswahl ist allerdings begrenzt. Kann das gewünschte Verhalten nicht über einen vordefinierten Actor Filter abgebildet werden, gibt es die Möglichkeit individuelle Actor Filter zu programmieren.
Bonitasoft erzielt auch für das 2. Kriterium 2 Punkte, da für den verwendeten Actor Filter Code genutzt werden muss. Der Zuordnungsmechanismus setzt sich hier aus der Variable, einer Belegung der Variable und dem Actor Filter zusammen, welche alle nativ in Bonitasoft zur Verfügung gestellt werden. Zur Entwurfszeit muss im Business Data Modell in der „bom.xml"-Datei ein Datentyp für die Variable angelegt werden. Danach kann in dem „Data"-Tab des Pools in dem Abschnitt „Pool variables" eine Pool Variable erstellt werden, die später mit dem Namen des Users belegt werden soll. Um die Variable zu belegen, kann entweder ein Formular oder ein Skript verwendet werden. Da ein Formular nativ verfügbar ist und ein Skript unter die Kategorie „Logik programmiert" fallen würde, wird die Formular-Option betrachtet. Im Formular kann nur der Datentyp aus dem Business Data Modell belegt werden, indem dieser als Input für die Task definiert wird. Dies erfolgt im „Execution"-Tab der Task unter „Contract" und dort unter „Input". Hier wird per Knopfdruck ein Input aus dem Datentyp erstellt. Zusätzlich wird in dem „Execution"-Tab unter dem Punkt „Operations" eine Operation angelegt, mit der dieser Input auf eine Variable zugewiesen werden kann. Als Variable muss nun die Pool Variable angegeben werden. Nachdem diese Schritte erfolgreich durchlaufen wurden, kann ein Formular im „Execution"-Tab unter „Form" angelegt werden. Dieses wird auf Knopfdruck samt sämtlicher vorher definierter Inputs automatisch angelegt. Pro definiertem Input verfügt das Formular somit über ein Eingabefeld. Aufgrund der Operation wird die Eingabe eines

solchen Feldes auf die Pool Variable zugewiesen, welche in einer anderen Task im Actor Filter verwendet werden kann. Auf diese Weise wird die angelegte Variable durch das Formular belegt und im Actor Filter als Input genutzt, um einen User über seinen Namen zu suchen. Das Belegen einer Variable durch ein Formular ist also deutlich komplexer als beispielweise in Axon Ivy.
Falls ein Actor Filter keinen passenden User finden sollte, kann dies nicht nativ überprüft oder abgefangen werden. Dies kann nur durch individuellen Code erfolgen, welcher die Logik beinhaltet. Daher erhält Bonitasoft für das 3. Kriterium 0 Punkte. Die Überprüfung müsste also im Input des Actor Filters erfolgen.
Abschließend erzielt Bonitasoft für das Pattern „Deferred Distribution“ die finale Punktzahl 4 von 10.

Für das 1. Kriterium erhält Camunda 4 Punkte, da das Erstellen einer Variable und eines Formulars nativ im Tool erfolgt. Die Identifizierung der Ressource kann auch in Camunda auf die Laufzeit verschoben werden, indem eine Variable verwendet oder zur Entwurfszeit keine Ressource in der Task angegeben wird. Bei letzterem kann nur der Administrator die Workitems der Task sehen und diese zur Laufzeit einem gewünschten User zuordnen. Falls eine Variable verwendet wird, muss der Name der Variable mit in „${…}“ hinterlegt werden. Durch die Zeichenfolge „${…}“, erkennt Camunda, dass es sich bei allem innerhalb der geschwungenen Klammern, nicht um einen String, sondern um eine Variable handelt. Die Variable kann entweder durch ein Formular oder ein Skript belegt werden. Wie bei den anderen Tools, wurde auch hier die Möglichkeit der Belegung durch das Formular betrachtet.
Der Zuordnungsmechanismus wird durch die Variable und das Formular abgebildet, welche standardmäßig in Camunda vorhanden sind, wodurch das Tool für das 2. Kriterium 4 Punkte erhält. In einer User Task kann im „Forms“-Tab „Generated Task Form“ ausgewählt werden, sodass Camunda automatisch ein Formular erstellt. Anschließend muss unter „Form Fields“ hinterlegt werden, welche Eingabefelder das Formular haben soll. Hier kann der Name bzw. die ID des Feldes frei gewählt werden. Alternativ gibt es auch die Möglichkeit, ein Formular per Drag-and-Drop selbst zu erstellen. Hierbei gibt es eine größere Auswahl an Elementen, aber die Verknüpfung zwischen Formular und Task muss manuell erfolgen, was zu Problemen führen kann. Die ID bzw. der Name des Eingabefeldes kann anschließend in der gewünschten Task unter „Assigee“ oder auch „Candidate Users“ folgendermaßen angegeben werden: „$(NameEingabefeld)“.
In dem 3. Kriterium erzielt Camunda 0 Punkte, da keine native Möglichkeit vorhanden ist, eine Variable bzw. den auf der Variable gespeicherten User zu überprüfen. Die benötigte Logik muss also individuell programmiert werden.
Zusammengerechnet schließt Camunda das Pattern „Deferred Distribution“ mit der Punktzahl 8 von 10 ab.

Für das Pattern „Authorization“ wurden zwei Kriterien überprüft:

- 1. Die Möglichkeit, den Umfang der Privilegien festzulegen, die eine Ressource in Bezug auf die Ausführung eines Prozesses besitzt. Diese Privilegien sind unabhängig von der eigentlichen Arbeitsverteilung und definieren den Umfang an Aktionen, die eine Ressource ausführen kann, wenn sie Workitems übernimmt
    - Gewichtung: *1
- 2. Es kann deklariert werden, was passieren soll, wenn die passende Ressource nicht die benötigten Privilegien hat und so ggf. die Workitems der Task nicht ausführen kann
    - Gewichtung: *0,5

Bei diesem Pattern werden keine konkreten Genehmigungen überprüft und auch keine Angaben zum Umfang der Privilegien gemacht. Dies erschwert die Bewertung, weswegen hier in erster Linie die Anzahl und der Anwendungsbereich der Genehmigungen betrachtet wird.

Axon Ivy erhält für das 1. Kriterium nur 3 Punkte, da Genehmigungen nativ vorhanden sind und dies auch umfangreich, allerdings sind diese nicht so flexibel einsetzbar, wie

z.B. in Camunda. In Axon Ivy werden Aspekte wie Genehmigungen und Sichtbarkeit von GUI-Elementen durch die Rollen und User im Organisationsmodell abgedeckt [15]. Zusätzlich können in der „Admin“-Weboberfläche Usern und Rollen sog. Permissions zugeordnet werden. Die Liste an Genehmigungen ist umfangreich und verschachtelt, eine Auflistung sämtlicher Genehmigungen konnte in der Dokumentation allerdings nicht gefunden werden. Diese können nicht alle im Detail betrachtet werden, aber es sind verschiedene Genehmigungen möglich, wie z.B. ob ein User bestimmte Workitems sehen, beenden oder neu zuordnen kann. Allerdings sind diese Genehmigungen eher statisch und immer nur allgemein für alle Prozesse gültig. Es gibt keine Möglichkeit, eine Genehmigung nur für einen Prozess, eine Task oder sogar ein Workitem zu vergeben, wie es z.B. in Camunda möglich ist.
Im 2. Kriterium erzielt Axon Ivy 0 Punkte, da die Überprüfung der Genehmigungen eines Users standardmäßig nicht möglich ist und nur mithilfe einer API umgesetzt werden kann [12]. Das Verhalten für den Fall, dass der User die benötigte Genehmigung nicht besitzt, muss individuell programmiert werden.
Für das Pattern „Authorization“ erzielt Axon Ivy somit final die Punktzahl 5 von 10.

Bonitasoft bildet Genehmigungen durch das Organisationsmodell in Verbindung mit sog. Profiles ab [36]. Das Organisationsmodell wird genutzt, um User, Rollen und Gruppen zu erstellen, welche später zur Zuordnung der Workitems genutzt werden. Die Profiles beinhalten die eigentlichen Genehmigungen. Hierbei gibt es allerdings nur zwei Profile: User und Administrator [36]. Um eigene Profile mit entsprechenden Genehmigungen anzulegen, muss die bezahlte Version von Bonitasoft verwendet werden (s. digitaler Anhang - Beantwortet) [36]. Abgesehen von den zwei Profilen, User und Administrator, welche nicht bearbeitet werden können, gibt es somit in der Community Version keine Möglichkeit, Usern Privilegien zuzuteilen bzw. diese zu bearbeiten (s. digitaler Anhang – Beantwortet) [36].
Demnach erhält Bonitasoft für das gesamte Pattern „Authorization“ die Punktzahl 0 von 10. Hier wurde also ein Aspekt, welcher in BPMN eigentlich vorhanden ist [93], hinter eine Paywall verschoben.

Camunda erhält für das 1. Kriterium 4 Punkte, da es standardmäßig in der „Admin“-Weboberfläche möglich ist, viele verschiedene und flexibel einsetzbare Genehmigungen zu erstellen. In der „Admin“-Weboberfläche von Camunda können unter dem Punkt „Authorizations“ die verschiedenen Genehmigungen eingestellt werden. Es liegen 19 verschiedene Bereiche vor, in denen umfangreiche Genehmigungen vergeben werden können. Jede Genehmigung kann entweder für eine Gruppe oder einen einzelnen User erstellt werden. Diese können sowohl statisch für die gesamte Applikation als auch flexibel für einzelne Prozesse, Tasks oder sogar laufende Workitems vergeben werden. Es ist nicht möglich, auf alle Genehmigungen einzugehen, daher werden im Folgenden für ein besseres Verständnis beispielhaft einzelne Berechtigungen benannt: Für eine Task können z.B. sowohl die „Create“-, „Read“-, „Update“-, „Delete“-Berechtigung (CRUD-Berechtigungen) als auch „TASK_ASSIGN“ oder „TASK_WORK“ vergeben werden. Bei einem Prozess können z.B. „UPDATE_TASK“ oder auch „DELETE_INSTANCE“ und „READ_HISTORY“ vergeben werden. Sämtliche Berechtigungen können von einem Administrator sowohl zur Entwurfszeit als auch zur Laufzeit ergänzt, verändert oder entfernt werden.
Für das 2. Kriterium erhält Camunda 0 Punkte, da die Logik des Kriteriums programmiert werden muss und Camunda nur eine API zur Verfügung stellt, um die Berechtigungen eines Users zu überprüfen [40]. Eine native Lösung für das 2. Kriterium ist nicht vorhanden.
Dementsprechend erreicht Camunda bei dem Pattern „Authorization“ die Punktzahl 6,67 von 10.

Das nächste Pattern, welches untersucht wurde, ist „Separation of Duties“. Bei diesem wurden drei Kriterien betrachtet:

- 1. Die Möglichkeit festzulegen, dass Workitems von zwei Tasks in einem konkreten Case von unterschiedlichen Ressourcen ausgeführt werden müssen
    - Gewichtung: *1
- 2. Es kann das Separation-of-Duties-Verhältnis zwischen einer Task „t“ und vorherigen Tasks abgebildet werden, wodurch keine Ressource (können auch mehrere Ressourcen sein, wenn es ein Loop gab), welche Workitems der vorherige Tasks ausgeführt hat, die Workitem der Task „t“ ausführen kann
    - Gewichtung: *1
- 3. Es kann deklariert werden, was passieren soll, wenn der Zuordnungsmechanismus keine passende Ressource identifiziert, da aufgrund des Separation-of-Duties-Verhältnisses keine passenden Ressourcen mehr übrig sind
    - Gewichtung: *0,5

Für alle drei Kriterien erzielt Axon Ivy jeweils 0 Punkte, da die entsprechende Logik programmiert werden muss (s. digitaler Anhang – Beantwortet). Nativ existieren nur drei Möglichkeiten, Ressourcen für die Workitems einer Task festzulegen, durch die Felder: „Role“, „Role from Attr.“, „User from Attr.“. Bei „Role“ kann nur aus einem Dropdown eine Rolle ausgewählt werden und bei „Role from Attr.“ bzw. „User from Attr.“ kann immer nur eine Rolle bzw. ein User hinterlegt werden. Es gibt keine Möglichkeit, User auszuschließen, die Workitems einer vorherigen Task ausgeführt haben. Dies ist nur durch individuellen Code für die entsprechenden Logik umsetzbar (s. digitaler Anhang – Beantwortet). Ein Negationsoperator, um User auszuschließen, wie er in BPMN durch XPath vorhanden ist [93], existiert in Axon Ivy nicht. Dementsprechend kann auch nur durch individuell programmierte Logik mit dem Fall umgegangen werden, dass keine passende Ressource übrig ist.
Daraus ergibt sich die finale Punktzahl 0 von 10 für Axon Ivy bei dem „Separation of Duties“-Pattern.

Für das 1. und 2. Kriterium erhält Bonitasoft jeweils 0 Punkte, da diese nur durch einen benutzerdefinierten Actor Filter umgesetzt werden können. Um diese Kriterien umzusetzen, muss die Zuordnung der Ressourcen variabel erfolgen. Es muss überprüft werden, welche User die Workitems der entsprechenden vorherigen Task ausgeführt haben, um sie von der Ausführung der Workitems der aktuellen Task auszuschließen. Für eine variable Zuordnung müssen in Bonitasoft Actor Filter verwendet werden (s. digitaler Anhang – Öffentlich). Keiner der vordefinierten Actor Filter bildet das Separation-of-Duties-Verhalten ab [25]. Der „Single user“-Filter kann die Workitems variabel einem bestimmten User zuteilen. Hier müsste aber die gesamte Logik des Patterns in dem Input des Filters programmiert werden, da dieser nur die ID eines Users annimmt. Ein „Task performer“-Filter teilt die Workitems einer Task dem User zu, der Workitems einer vorherigen Task abgeschlossen hat. Als Input benötigt er den Namen der vorherigen Task. Es gibt allerdings keine Möglichkeit, diesen User von einer Menge an potenziellen Usern auszuschließen. Da diese Standardfilter nicht verändert werden können, müsste ein eigener Filter geschrieben werden, der die gesamte Logik des Patterns abbildet (s. digitaler Anhang – Beantwortet).
Dementsprechend wird das 3. Kriterium in Bonitasoft auch mit 0 Punkten bewertet, da ein solches Verhalten im benutzerdefinierten Actor Filter hinterlegt sein muss (s. digitaler Anhang – Beantwortet) und daher nur individuell programmiert werden kann.
Somit erzielt Bonitasoft im Pattern „Separation of Duties“ die Punktzahl 0 von 10. Auch in Bonitasoft fehlt, anders als in BPMN durch XPath [93], ein Negations- bzw. Ausschlussoperator.

Camunda erhält für die Kriterien 1. und 2., jeweils 0 Punkte, da das gewünschte Verhalten nur durch programmierte Logik abgebildet werden kann (s. digitaler Anhang – Öffentlich). Weder in dem Feld „Assignee“ noch in „Candidate Users“ oder „Candiate Groups“ können User ausgeschlossen werden. Ein solcher Operator, wie er in BPMN durch XPath möglich ist [93], fehlt hier. Daher muss das Verhalten individuell programmiert werden. Für diese Umsetzung müssen Genehmigungen, Variablen, Historische-Daten etc. angelegt und verwendet werden (s. digitaler Anhang – Öffentlich).

Auch das 3. Kriterium wird daher in Camunda mit 0 Punkten bewertet, da ein solches Verhalten in der entsprechenden individuell programmierten Logik für die ersten beiden Kriterien hinterlegt sein muss.
Auch Camunda erzielt in dem Pattern „Separation of Duties“ somit die Punktzahl 0 von 10.

Die folgenden zwei Bewertungskriterien wurden für das Pattern „Case Handling“ untersucht:

- 1. Die Möglichkeit, alle Workitems innerhalb eines bestimmten Falls (Case) derselben Ressource zuzuordnen, die den Fall gestartet hat
    - Gewichtung: *1
- 2. Zu Beginn des Prozesses wird eine Ressource ausgewählt (aus einer Menge von Ressourcen oder aus einer Rolle), welche dann alle Workitems dieses Cases erhält
    - Gewichtung: *1

Es gibt zwei Varianten, das Pattern umzusetzen: „Hard“ oder „Soft“. Für die „Hard“-Variante muss der gewählte User die Workitems ausführen und darf sie nicht mehr abgeben [93]. Um die „Soft“-Variante umzusetzen, erhält der User zwar die Workitems, darf diese aber auch anderen Usern zuordnen und muss sie nicht zwingend selbst ausführen [93].

Das 1. Kriterium kann in Axon Ivy mit 4 Punkten bewertet werden, da das gewünschte Verhalten durch die vordefinierten Rollen „CREATOR“ oder auch „SELF“ möglich ist. Diese Rollen sind standardmäßig verfügbar und können in jeder beliebigen Task unter „Responsible“ als Rolle ausgesucht werden. Falls das Verhalten gewünscht ist, dass der User, welcher den Prozess gestartet hat, sämtliche Workitems des Cases übernimmt, muss in jeder User Task die Rolle „CREATOR“ ausgewählt werden. Sollte der Prozess von außen gestartet werden, z.B. per API, und der User, der das erste Workitem des Prozesses abschließt, ist definiert als User, welcher den Fall gestartet hat, so können alle anderen Tasks des Prozesses die Rolle „SELF“ erhalten. Auf diese Weise wird immer der User gewählt, der das Workitem der vorherigen Task ausgeführt hat. Ein Workitem der ersten Task kann dann ganz normal durch das „Responsible“-Feld angeboten werden oder von einem Administrator zugeordnet werden.
Axon Ivy erhält im 2. Kriterium 3 Punkte, da das Tool Workitems nur einer Rolle oder einem einzelnen User anbieten kann, wie in „Direct Distribution“ beschrieben. Eine Menge von einzelnen Ressourcen ist ohne benutzerdefinierten Code nicht möglich, weswegen hier nicht die volle Punktzahl vergeben werden kann, obwohl das restliche Verhalten, wie oben beschrieben, nativ möglich ist.
Abschließend erhält Axon Ivy für das Pattern „Case Handling“ die Punktzahl 8,75 von 10. Dies ist vor allem interessant, da dieses Pattern in BPMN gar nicht abbildbar sein soll [93]. Allerdings kann dies angezweifelt werden, da [93] außerdem aussagt, dass das anschließende Pattern „Retain Familiar“ möglich ist und „Case Handling“ umgesetzt werden kann, indem „Retain Familiar“ auf jede Task angewandt wird. Für die „Hard“-Variante des Patterns muss die „Delegate“-Funktion durch die Genehmigungen eingeschränkt sein. Falls dies nicht der Fall ist, ist die „Soft“-Variante umgesetzt, da der User zwar die Workitems erhält, diese aber nicht ausführen muss und über die „Delegate“-Funktion auch anderen Usern zuordnen kann.

Bonitasoft erfüllt das 1. Kriterium mit 4 Punkten, da die zwei Actor Filter „Task performer“ und „Initiator“ standardmäßig vorhanden sind. Für den Fall, dass der User, welcher den Prozess gestartet hat, sämtliche Workitems des Cases übernimmt, kann der Actor Filter „Initiator“ in allen Tasks des Prozesses gewählt werden. Falls der User, welcher das erste Workitem des Prozesses abschließt, für den Case verantwortlich sein soll, kann in allen folgenden Tasks „Task performer“ als Actor Filter gewählt und der Name der ersten Task kann als Input mitgegeben werden. Ein Workitem der ersten Task kann dann normal durch das Actor Mapping angeboten werden.
Auch für das 2. Kriterium erhält Bonitasoft 4 Punkte. Wie oben beschrieben kann ein Workitem der ersten Task durch das Actor Mapping angeboten werden. Das Actor

Mapping verfügt über die Möglichkeit, Workitems nativ einem oder mehreren Usern oder Rollen anzubieten.
Das finale Ergebnis von Bonitasoft für das Pattern „Case Handling“ lautet somit 10 von 10. Dies ist, aus dem gleichen Grund wie in dem Abschnitt über Axon Ivy, besonders hervorzuheben. Eine „Soft“-Implementierung des Patterns ist in Bonitasoft nur im Rahmen der „Release“-Funktion möglich. Hierbei kann ein Workitem, nachdem dieses zuvor angenommen (claim) wurde, wieder freigegeben werden. Eine Möglichkeit das Workitem anderen Usern zuzuordnen, liegt nicht vor. Für eine „Hard“-Variante des Patterns müsste eingeschränkt werden, dass ein Workitem nicht mehr freigegeben werden darf, nachdem es allokiert wurde. Aufgrund der fehlenden Genehmigungen ist dies in der Community Version nicht möglich.

Für das 1. Kriterium erhält Camunda 2 Punkte, da das Kriterium teilweise programmiert werden muss, um es umzusetzen. Soll der Initiator des Cases alle Workitems erhalten, so kann im Start Event des Prozesses eine Variable hinterlegt werden, auf welcher der Initiator gespeichert wird. Diese Variable kann anschließend auf folgende Weise im „Assignee“-Feld in sämtlichen User Tasks des Prozesses hinterlegt werden: „${variableName}“. Sollte der User, welcher das erste Workitem abschließt, gewünscht sein, muss mit einem Task Listener gearbeitet werden (s. digitaler Anhang – Öffentlich). Hier kann ein Workitem der ersten Task eines Prozesses beliebigen Usern oder Gruppen unter „Candidate Users“ bzw. „Candidate Groups“ offeriert werden oder von einem Administrator einem bestimmten User direkt zugeteilt werden. In dieser Task wird ein Task Listener hinterlegt, welcher beim Abschluss eines Workitems der Task folgende Code-Zeile ausführt: „${task.execution.setVariable('taskCompletedBy', task.assignee)}“. Auf diese Weise wird die Variable „askCompletedBy“ erstellt und der entsprechende User darauf abgespeichert. Diese Variable „${'taskCompletedBy'}“ kann nun in sämtlichen User Tasks des Prozesses im „Assignee“-Feld eingetragen werden. Auch wenn die erste Möglichkeit ohne Code umsetzbar ist, wird Code für die zweite Variante benötigt. Da für dieses Kriterium allerdings beide Varianten betrachtet werden, ergibt sich die entsprechende Punktzahl.
Camunda kann das 2. Kriterium standardmäßig umsetzen und erhält 4 Punkte. Durch die Felder „Candidate Users“ und „Candidate Groups“ kann ein Workitem der ersten Task sowohl mehreren Ressourcen einzeln als auch mehreren Gruppen angeboten werden. Auch eine Kombination aus beidem ist möglich.
Camunda erzielt somit für das Pattern „Case Handling“ die Punktzahl 7,5 von 10. Auch bei Camunda ist dies, aus dem gleichen Grund wie in dem Abschnitt über Axon Ivy, besonders hervorzuheben. Die „Hard“-Variante des Patterns ist durch die Standard-Authorization „TASK_WORK“ umgesetzt, wodurch User ein Workitem nur bearbeiten, aber dieses nicht erneut zuordnen können, außer sie erhalten explizit die Berechtigung dafür. Für die „Soft“-Variante muss die Standard-Authorization auf „TASK_UPDATE“ geändert werden, wodurch die User ein Workitem nicht nur bearbeiten, sondern auch neu zuordnen können.

Bei dem Pattern „Retain Familiar“ wurden die folgenden beiden Bewertungskriterien untersucht:

- 1. Die Möglichkeit, ein Workitem der gleichen Ressource zuzuordnen, die ein vorhergehendes Workitem für denselben Fall übernommen hat
    - Gewichtung: *1
- 2. Das Pattern wird durch eine Beziehung zwischen einer Task und einer beliebigen vorherigen Task abgebildet
    - Gewichtung: *1

Sowohl das 1. als auch das 2. Kriterium erhalten in Axon Ivy jeweils 2 Punkte, da eine Zeile Code benötigt wird, um den Usernamen der Ressource zu erhalten, welche den Fall gestartet hat bzw. das Workitem der ersten Task im Fall ausgeführt hat. Anders als im vorherigen Pattern „Case Handling“ kann hier nicht die Rolle „SELF“ verwendet werden, da diese nur den User des Workitems der direkt vorangegangenen Task findet. Bei

diesem Pattern soll eine beliebige vorherige Task gewählt werden können. Daher wird hier neben der Zeile Code zusätzlich eine Variable benötigt, auf welcher der Username abgespeichert wird und ein User Dialog, damit die Variable belegt werden kann. Dies ist zwar nativ möglich, aber aufgrund der Zeile Code können die Kriterien nur mit 2 Punkten bewertet werden. Hierfür muss zuerst in der Datei „Data“ eine entsprechende Variable angelegt werden, z.B. „userName“. Anschließend kann eine beliebige vorherige Task des Prozesses ausgewählt werden. Ein Workitem dieser Task kann einem User oder einer Rolle angeboten werden. Zusätzlich erhält diese Task einen User Dialog, in welchem die Variable als Input und Output, nicht aber als UI-Element, angegeben wird. Axon Ivy generiert anschließend den User Dialog automatisch. Der User Dialog wird benötigt, da ansonsten der „Output“-Tab, in dem die Variable belegt wird, nicht ausgeführt wird. Dieser Tab wird nur verwendet, wenn ein User Dialog hinterlegt ist, auch wenn dieser keine Felder als GUI-Elemente beinhaltet, also leer ist. Im „Output“-Tab wird die Variable in der Spalte „Attribute“ angezeigt und der Befehl „ivy.task.getWorkerUserName()“ kann in die entsprechenden Zeile der Spalte „Expression“ eingesetzt werden. Auf diese Weise wird der Name des Users auf der Variable abgespeichert, welcher ein Workitem der ausgewählten Task ausgeführt hat. Dies muss im „Output“-Tab erfolgen, da nur der Username des Users gespeichert werden soll, der das Workitem auch wirklich abschließt. In einer beliebigen nachfolgenden Task kann nun unter „Responsible“ bei „User from Attr.“ der Befehl „in.userName“ eingetragen werden. Ein Workitem dieser Tasks wird somit dem User angeboten, der das Workitem der vorherigen beliebigen Task ausgeführt hat.
Zusammengefasst erhält Axon Ivy somit für das Pattern „Retain Familiar“ als Endergebnis 5 von 10 Punkte.

In Bonitasoft können beide Kriterien standardmäßig abgebildet werden und erhalten somit jeweils 4 Punkte. Durch den Actor Filter „Task performer“ können die Workitems einer Task dem gleichen User zugeordnet werden, der die Workitems einer beliebigen vorherigen Task ausgeführt hat. Hierfür muss nur der Name einer beliebigen vorherigen Task dem Actor Filter als Input gegeben werden. Der Actor Filter wird in einer Task hinterlegt und kann den Namen einer beliebigen vorherigen Task als Input annehmen.
Bei dem Pattern „Retain Familiar“ erzielt Camunda also abschließend die Punktzahl 10 von 10.

In Camunda können beide Kriterien durch einen Task Listener und eine Zeile Code umgesetzt werden, wodurch die Kriterien jeweils 2 Punkte erhalten. Dasselbe Vorgehen wie in „Case Handling“ wird auch hier verwendet. In der gewünschten vorherigen Task wird ein Task Listener hinterlegt, der zum Abschluss eines Workitems der Task die folgende Zeile Code ausführt: „${task.execution.setVariable('taskCompletedBy', task.assignee)}“. Auf der Variable „taskCompletedBy“ ist danach der User gespeichert, welcher ein Workitem der Task ausgeführt hat. Diese Variable kann in dem „Assignee“-Feld einer beliebigen nachfolgenden Task hinterlegt werden.
Camunda erhält somit insgesamt die Punktzahl 5 von 10 für das Pattern „Retain Familiar“.

Die nachfolgenden vier Kriterien wurden für das Pattern „Capability-Based Distribution“ betrachtet:

- 1.1 Die Fähigkeit, Workitems Ressourcen zuzuordnen, basierend auf spezifischen Fähigkeiten, die die Ressourcen besitzen
    - Gewichtung: *1
- 1.2 Fähigkeiten (und die damit verbundenen Werte) werden für einzelne Ressourcen als Teil des Organisationsmodells erfasst
    - Gewichtung: *1
- 2. Jede Task kann eine Capability-Function (Abfragen ggf. in Programmiersprachen) haben, welche die Ressourcen liefert, die die benötigten Fähigkeiten für die Task besitzen
    - Gewichtung: *1

- 3. Capabilities können als Schlüssel-Wert-Paar definiert werden
    - Gewichtung: *0,5

Das Pattern kann auf zwei verschiedene Arten implementiert werden: „Push“- oder „Pull“-Variante [93]. Bei der „Pull“-Variante kann die Ressource Workitems suchen, welche sie aufgrund ihrer Fähigkeiten ausführen kann [93]. Bei der „Push“-Variante sucht das System passende Ressourcen für ein Workitem aus und ordnet das Workitem diesen zu [93]. Für eine „Pull“-Variante müsste eine entsprechende Funktionalität oder ein GUI-Element für die User vorliegen. Dies ist in keinem Tool der Fall. Dementsprechend müsste für diese Variante nicht nur die Logik, sondern auch GUI-Elemente bzw. ein passendes Frontend entwickelt werden. Diese Variante würde somit in den entsprechenden Kriterien mit 0 Punkten bewertet werden. Die „Pull“-Variante wurde daher nicht weiter betrachtet, da es realistischer ist die „Push“-Variante in den Tools umzusetzen und eine Variante das Pattern ausreichend repräsentiert.

Axon Ivy erhält für die Kriterien 1.1 und 2. jeweils 0 Punkte, da auf Fähigkeiten der User nur per Code, hier IvyScript genannt, zugegriffen werden kann [15]. Die Fähigkeiten werden in Axon Ivy als Additional properties einem User hinzugefügt und können nur über IvyScript genutzt werden [15]. Hierfür muss eine entsprechende API eingesetzt werden, welche den Wert der Fähigkeiten liefert [11]. Der Zuordnungsmechanismus muss demnach ebenfalls individuell programmiert werden.
Für die Kriterien 1.2 und 3. erhält Axon Ivy jeweils 4 Punkte, da es möglich ist, Fähigkeiten als Schlüssel-Wert-Paar für User im Organisationsmodell zu hinterlegen. Hier können in der „user.xml“-Datei pro User die Fähigkeiten unter dem Punkt „Additional properties“ eingefügt werden. Diese bestehen aus einem Schlüssel (z.B. „Abschluss“) und einem dazugehörigen Wert (z.B. „Luftfahrtwesen“).
Axon Ivy erhält abschließend die Punktzahl 4,29 von 10 für das Pattern „Capability-Based Distribution“. Die Umsetzung erfolgt nach dem „Push“-Prinzip, wobei das System die Fähigkeiten überprüft und das Workitem der passenden Ressource zuweist. Sollten mehrere passende Ressourcen gefunden werden, muss zusätzlich eine Auswahllogik implementiert werden. In BPMN ist dieses Pattern nicht umsetzbar, da es keinen Verteilungsmechanismus basierend auf zusätzlichen Attributen der User gibt [93].

Kriterium 1.1 und 2. erhalten in Bonitasoft jeweils 0 Punkte, da es standardmäßig nicht möglich ist, mehrere Fähigkeiten eines Users zu überprüfen und basierend darauf die Workitems zuzuordnen (s. digitaler Anhang – Unbeantwortet). Fähigkeiten können für jeden User in Bonitasoft unter „Custom User Information“ im „User“-Tab des Organisationsmodells hinterlegt werden. Bonitasoft verfügt zwar über den Actor Filter „Users with Custom information“, welcher nativ aus der potenziellen Usermenge einer Task, die durch Actor Mapping festgelegt wurde, die User findet, welche eine bestimmte Fähigkeit haben. Sollten mehrere User identifiziert werden, so bietet der Filter allen Usern das Workitem an. Allerdings kann dieser Filter immer nur eine Fähigkeit gleichzeitig überprüfen und weder mehrfach noch ineinander verschachtelt genutzt werden (s. digitaler Anhang – Unbeantwortet). Auch auf Nachfrage konnte keine Lösung hierfür gefunden werden (s. digitaler Anhang – Unbeantwortet). Demnach können nativ nicht mehrere Fähigkeiten überprüft werden, was das Pattern verlangt und näher an der Praxis liegt. Daher muss für diese beiden Kriterien ein eigener Actor Filter geschrieben werden, der die Logik des Patterns umsetzt. Auch der Fall, dass mehrere passende User gefunden werden, muss bei dem benutzerdefinierten Actor Filter berücksichtigt werden.
Bonitasoft erhält in den Kriterien 1.2 und 3. jeweils 4 Punkte, da es möglich ist, Fähigkeiten nativ als Schlüssel-Wert-Paar für User im „User“-Tab des Organisationsmodells zu hinterlegen. Für sämtliche User werden erst im „User“-Tab die möglichen Schlüssel hinterlegt. Anschließend werden die Schlüssel für jeden User angezeigt und können individuell mit Werten belegt werden.
Bonitasoft erhält somit für das Pattern „Capability-Based Distribution“ die finale Punktzahl 4,29 von 10. Die Umsetzung erfolgt nach dem „Push“-Prinzip, wobei das System die Fähigkeiten überprüft und das Workitem der passenden Ressource zuweist. Sollten mehrere passende Ressourcen gefunden werden, muss eine zusätzliche Auswahllogik

im Actor Filter implementiert werden oder das Workitem wird allen Usern angeboten. Da dieses Pattern, wie im Axon Ivy Abschnitt beschrieben, nicht in BPMN umsetzbar ist [93], ist hier hervorzuheben, dass das Überprüfen einer Custom User Information in Bonitasoft sogar standardmäßig mithilfe des Actor Filters „Users with Custom information“ möglich ist.

Camunda erhält für die Kriterien 1.2 und 3. jeweils 0 Punkte, da es nicht möglich ist, den Usern Fähigkeiten oder zusätzliche Attribute mitzugeben (s. digitaler Anhang – Beantwortet). Die einzige Möglichkeit dafür besteht durch eine externe Verwaltung der User wie bei Lightweight Directory Access Protocol (LDAP) und/oder eine eigenerstellte Authentifizierung (s. digitaler Anhang – Beantwortet & Öffentlich). Camunda verfügt somit über keine eigenen Funktionen, um dieses Pattern nativ abzubilden.
Dementsprechend können auch die Kriterien 1.1 und 2. nicht umgesetzt werden und erhalten ebenfalls 0 Punkte. Diese beiden Kriterien können nicht abgebildet werden, da in Camunda keine Fähigkeiten vorhanden sind, welche zur Zuordnung der Workitems verwendet werden könnten.
Final erhält Camunda somit für das Pattern „Capability-Based Distribution“ die Punktzahl 0 von 10. Daher wurde weder die „Push“- noch die „Pull“-Variante des Patterns umgesetzt. Hier liegen die Funktionen von Camunda näher an denen von BPMN [93], als an denen der anderen beiden Tools.

Für das Pattern „History-Based Distribution“ wurden die folgenden drei Kriterien untersucht:

- 1. Die Möglichkeit, Workitems Ressourcen zuzuordnen, und zwar auf der Grundlage der bisherigen Ausführungshistorie der Ressourcen
    - Gewichtung: *1
- 2. Jede Task kann eine Historical-Distribution-Function (Abfragen ggf. in Programmiersprachen) haben, welche die Ressourcen liefert, die die benötigten Kriterien (beliebige Kombination von Anforderungen, die sich aus der Ausführungshistorie ermitteln lassen) bezüglich ihrer Ausführungshistorie für die Task erfüllen
    - Gewichtung: *1
- 3. Die Ausführungshistorie wird abgespeichert und kann verwendet werden
    - Gewichtung: *1

Dieses Pattern ist in BPMN möglich aufgrund der historischen Ausführungsdaten, welche durch den XPath Befehl „getTaskHistory“ geliefert werden [93]. In keinem der drei untersuchten Tools wird XPath verwendet. Axon Ivy und Camunda liefern eigene API-Befehle, um auf die historischen Ausführungsdaten zuzugreifen (s. digitaler Anhang - Beantwortet).

Axon Ivy erhält für das 1. und 2. Kriterium jeweils 0 Punkte. Die Funktionalitäten sind nicht standardmäßig im Tool umsetzbar und könnten nur durch programmierte Logik mithilfe von API-Funktionen für Abfragen, sog. Case- und Task-Queries, abgebildet werden (s. digitaler Anhang - Beantwortet).
Für das 3. Kriterium erzielt Axon Ivy 0 Punkte, da die Ausführungshistorie zwar abgespeichert wird, allerdings nicht nativ verwendet werden kann, sondern nur durch API-Funktionen für Case- und Task-Queries (s. digitaler Anhang - Beantwortet).
Insgesamt ergibt sich somit für Axon Ivy bei dem Pattern „History-Based Distribution“ die Punktzahl 0 von 10.

Alle drei Kriterien erhalten in Bonitasoft 0 Punkte, da es keine Möglichkeit gibt, auf die historischen Ausführungsdaten zuzugreifen, weder nativ noch durch eine API (s. digitaler Anhang - Öffentlich). Auch auf Nachfrage wurde keine Antwort gefunden (s. digitaler Anhang - Unbeantwortet).
Dementsprechend erzielt Bonitasoft für das Pattern „History-Based Distribution“ die Punktzahl 0 von 10.

Auch Camunda erhält für das 1. und 2. Kriterium jeweils 0 Punkte, da nur durch API-Funktionen auf die Ausführungshistorie zugegriffen werden kann, um die

Funktionalitäten umzusetzen (s. digitaler Anhang - Öffentlich) [41].
Das 3. Kriterium erhält in Camunda auch 0 Punkte, da die Ausführungshistorie automatisch abgespeichert wird, aber nicht nativ verwendet werden kann, sondern nur durch die API-Funktionen (s. digitaler Anhang - Öffentlich) [41].
Camunda erzielt die Punktzahl 0 von 10 für das Pattern „History-Based Distribution“.

Bei dem Pattern „Organisational Distribution“ wurden die folgenden drei Bewertungskriterien betrachtet:

- 1. Die Fähigkeit, Workitems Ressourcen zuzuordnen, basierend auf ihrer Position innerhalb der Organisation und ihren Beziehungen zu anderen Ressourcen
  - Gewichtung: *1
- 2. Es ist möglich, die Organisationsstruktur/Hierarchie und die Beziehungen unter den Ressourcen im Organisationsmodell abzubilden
  - Gewichtung: *1
- 3. Jede Task kann eine Organizational-Distribution-Function (Abfragen ggf. in Programmiersprachen) haben, welche die Ressourcen liefert, die die benötigten Kriterien (beliebige Kombination von Anforderungen, die sich aus dem Organisationsmodell ableiten lassen) bezüglich des Organisationsmodells erfüllen
  - Gewichtung: *1

In Axon Ivy wird das 1. und 2. Kriterium jeweils mit 3 Punkten bewertet, da die Hierarchie bzw. Organisationsstruktur standardmäßig abgebildet werden kann, aber die Möglichkeit fehlt, Beziehungen zwischen den Ressourcen umzusetzen. Wie bereits in „Role-Based Distribution“ erläutert, verfügt Axon Ivy über die zwei Rollen Konzepte: Permission Roles und Member Roles. Durch die Member Roles sollen Organisationsstrukturen abgebildet werden. Es ist möglich, die Member Roles in Hierarchien anzuordnen und in Verbindung miteinander zu bringen. Die Rollen können anschließend in einer User Task unter „Role“ bei „Responsible“ aus der Liste ausgewählt werden, sodass die Workitems nur den Usern angeboten werden, welche die passende Rolle besitzen. Wichtig ist hierbei anzumerken, dass die Workitems auch allen Rollen angeboten werden, welche in der Hierarchie als Unterrollen der in der Task hinterlegten Rolle festgelegt wurden. Eine genauere Beschreibung erfolgt im Pattern „Configurable Unallocated Work Item Visibility“. Allerdings fehlt Axon Ivy eine Möglichkeit, Beziehungen zwischen den Ressourcen abzubilden. Beispielsweise gibt es keine Möglichkeit anzugeben, dass ein User der Manager bzw. Vorgesetzte eines anderen Users ist und dementsprechend auch keine Möglichkeit, dies für die Zuordnung der Workitems zu verwenden, wie in Bonitasoft.
Auch für das 3. Kriterium erhält Axon Ivy 3 Punkte, da es mit dem „Responsible“-Feld in einer User Task die native Möglichkeit gibt, die Rollen und damit die erstellte Organisationsstruktur für die Zuordnung der Workitems zu verwenden. Allerdings kann bei dieser Möglichkeit immer nur eine Anforderung bezüglich der Organisationsstruktur überprüft werden. Es ist nicht möglich, eine Kombination von Anforderungen bezüglich des Organisationsmodells zu prüfen.
Abschließend erzielt Axon Ivy für das Pattern „Organisational Distribution“ die Punktzahl 7,5 von 10.

Bonitasoft erzielt bei dem 1. Kriterium 4 Punkte, da sich die Workitems nativ sowohl aufgrund der Organisationsstruktur als auch aufgrund der Beziehungen zwischen den Ressourcen zuordnen lassen. Wie in „Role-Based Distribution“ beschrieben, werden Rollen und Gruppen angelegt und den Usern zugewiesen. Nachdem ein Actor für die Swimlane oder für eine Task festgelegt wurde, kann dieser mit dem Actor Mapping belegt werden. Auf diese Weise wird festgelegt, welchen Usern zur Laufzeit die Workitems angeboten werden. Hier können einem Actor beliebige Entitäten aus dem Organisationsmodell zugewiesen werden:

- Der Actor wird einer oder mehreren Gruppe(n) zugewiesen
- Der Actor wird einer oder mehreren Rolle(n) zugewiesen
- Der Actor wird einer oder mehreren Membership(s) (bestimmte Rolle in bestimmter Gruppe) zugewiesen

- Der Actor wird einem oder mehreren User(n) zugewiesen

Außerdem kann der Actor einer beliebigen Kombination der oben genannten Möglichkeiten zugewiesen werden. Die Workitems werden anschließend sämtlichen Usern angeboten, welche mindestens ein zugewiesenes Kriterium besitzen. Ein User muss nicht alle Kriterien besitzen, um die Workitems zu erhalten. Zusätzlich zum Actor Mapping, welches Workitems aufgrund von Kriterien der Organisationsstruktur zuordnet, kann auch der Actor Filter „User manager“ verwendet werden, um Workitems aufgrund von Beziehungen zwischen den Ressourcen zuzuordnen. Bei diesem muss nur die ID eines Users als Input eingegeben werden und der Actor Filter wird die Workitems dem Manager dieses Users zuteilen. Auch wenn hierfür zwei Zeilen Code verwendet werden müssen, wie bei dem „Single user“ Actor Filter, erhält das Kriterium 4 Punkte, da alle anderen Aspekte dieses Kriteriums nativ im Tool möglich sind und das Kriterium in diesem Tool am besten umgesetzt werden kann.
Auch für das 2. Kriterium erhält Bonitasoft 4 Punkte, da standardmäßig sowohl die Organisationsstruktur als auch Beziehungen zwischen den Ressourcen in der „Organization.xml“-Datei abgebildet werden können. Neben Rollen können auch Gruppen angelegt werden, welche für Hierarchien miteinander verknüpft bzw. verschachtelt werden können. Zusätzlich können Memberships als Angabe einer konkreten Rolle in einer bestimmten Gruppe verwendet werden. Somit kann die Organisationsstruktur nativ durch Rollen, Gruppen und Memberships umgesetzt werden. Zusätzlich ist es möglich, in der User-GUI für jeden User einen Manager zu hinterlegen, um Beziehungen zwischen den Ressourcen darzustellen.
Bonitasoft erzielt im 3. Kriterium 3 Punkte, da es standardmäßig im Tool möglich ist, durch die Zuweisung der Actor auf die Tasks und dem anschließenden Actor Mapping die Ressourcen zu finden, welche die gewünschten Kombinationen von Kriterien bezüglich Rollen, Gruppen, Memberships erfüllen. Das Gleiche gilt für die Verwendung von Actor Filtern, um die Manager-Beziehung zwischen Usern zu überprüfen. Allerdings gibt es hier nicht die volle Punktzahl, da für den Actor Filter zwei Zeilen Code erforderlich sind und nur geprüft werden kann, wer der Manager eines Users ist. Es gibt keine Möglichkeit in den Standardfiltern, Unterstellte eines Managers zu finden [25]. Auch können in keinem Standardfilter mehrere Kriterien bezüglich der Beziehung zwischen Ressourcen überprüft werden [25], weswegen für ein solches Verhalten ein individueller Actor Filter programmiert werden müsste, wie in „Separation of Duties“ beschrieben.
Zusammenfassend erzielt Bonitasoft für das Pattern „Organisational Distribution“ die Punktzahl 9,17 von 10.

Camunda erhält für alle drei Kriterien 0 Punkte, da es keine Möglichkeit gibt, eine Hierarchie bzw. Organisationsstruktur und Beziehungen zwischen Ressourcen abzubilden (s. digitaler Anhang – Beantwortet). Dementsprechend ist es auch nicht möglich, auf Grundlage dessen Workitems den Ressourcen zuzuordnen (s. digitaler Anhang – Beantwortet). Gruppen können, wie in „Role-Based Distribution“ beschrieben, erstellt werden, allerdings gibt es keine standardmäßige Möglichkeit in Camunda, diese miteinander in Beziehung zu setzen bzw. hierarchisch anzuordnen. Auch ist keine Möglichkeit bei der Usererstellung/-bearbeitung vorhanden, um User miteinander in Beziehung zu setzen (s. digitaler Anhang – Beantwortet). Bei Fragen zu diesem Thema wird auf ein LDAP-Plugin von Camunda verwiesen (s. digitaler Anhang – Öffentlich).
Daher erhält Camunda abschließend die Punktzahl 0 von 10 für das Pattern „Organisational Distribution“.

Für das Pattern „Automatic Execution“ wurden die folgenden beiden Kriterien betrachtet:

- 1. Die Möglichkeit, dass ein Workitem einer Task ausgeführt werden kann, ohne die Dienste einer Ressource zu nutzen
    - Gewichtung: *1
- 2. Das Workitem der Task wird sofort gestartet, nachdem der Prozess die Task erreicht und nach dem Ausführen wird direkt das Workitem der nächsten Task

gestartet, ohne dass das ausgeführte Workitem jemals einer Ressource zugeordnet wurde
- Gewichtung: *1

Bei diesem Pattern soll überprüft werden, wie gut ein Workitem ausgeführt werden kann, ohne dass es einem User zugeordnet wird. Hierbei ist zu beachten, dass Workitems, welche keinem User zugeordnet werden, ihre Aufgabe selbstständig ausführen sollen, was in der Praxis bedeutet, dass Code verwendet werden muss. Einem Workitem der Task muss durch Code bzw. ein Skript vorgegeben werden, welche Aufgabe es selbstständig ausführen soll. Dementsprechend wird bei diesem Pattern nicht untersucht, ob Code verwendet wird, da dies in der Natur des Patterns liegt, sondern, ob es möglich ist, automatische Tasks zu erstellen und wenn ja, wie einfach dies ist. Ein Kriterium kann also 4 Punkte erhalten, wenn es besonders einfach ist bzw. es besonders viele Möglichkeiten gibt, automatische Tasks zu erstellen.

Axon Ivy erhält für beide Kriterien 4 Punkte, da es viele verschiedenen vordefinierte Möglichkeiten gibt die Workitems einer Task ohne Ressource ausführen zu lassen. Grundsätzlich können selbst Workitems einer User Task ohne eine Ressource ausgeführt werden, indem bei „Responsible“ unter „Role“ die Rolle „SYSTEM“ ausgeführt wird. Entsprechende Skripts, die ein Workitem der Task ausführen soll, können in den Code-Feldern der einzelnen Tabs hinterlegt werden. Auch ist eine Skript Task verfügbar, dessen Workitems ohne Zuordnung an eine Ressource ein hinterlegtes Skript ausführen können. Des Weiteren können sämtliche sog. Interface Activities automatisch ohne eine zugeordnete Ressource ausgeführt werden. Diese sind bereits auf eine bestimmte Schnittstelle oder Funktionalität vorprogrammiert und deren Workitems werden automatisch vom System ausgeführt. Zum Beispiel gibt es somit standardmäßige Interface Activities, um mit Datenbanken oder Webservice-Schnittstellen zu kommunizieren oder E-Mails zu versenden.
Abschließend erzielt Axon Ivy daher im Pattern „Automatic Execution“ die Gesamtpunkzahl 10 von 10.

Für die beiden Kriterien erhält Bonitasoft jeweils 3 Punkte, da es standardmäßig möglich ist, die Workitems einer Task automatisch ausführen zu lassen, aber keine vordefinierten Typen, wie in Axon Ivy, vorliegen. Darüber hinaus können Skripte immer nur in bestimmten Kontexten, wie z.B. Konnektoren oder Variablen, eingepflegt werden. Die Workitems aller Task-Arten bis auf Human Task werden automatisch ausgeführt. Die Workitems der Task-Arten Receive, Send und Call führen automatisch Aktionen im Workflow-Kontext aus, wie eine Nachricht an andere Pools senden oder einen anderen Prozess starten. Um eigene Skripte zu hinterlegen bzw. externe Aktionen anzusprechen, können die Task-Arten Abstract, Service und Script verwendet werden. Hier können Skripte entweder im Kontext der Konnektoren eingepflegt werden, um externe Systeme anzusprechen, oder im Kontext der Variablen, um diese zu verändern. Es existieren Konnektoren für LDAP, Representational State Transfer (REST), Web-Services, Datenbanken, Enterprise Resource Planning (ERP) Systeme, E-Mail und Groovy Script. Diese müssen aber vollkommen in Skripten konfiguriert werden, es gibt keine GUIs wie in Axon Ivy. Um Variablen zu belegen und zu verändern, können konkrete Skripte hinterlegt werden bei den Feldern „Operation“ und „Variabel“.
Bonitasoft erzielt somit zusammengerechnet für das Pattern „Automatic Execution“ den Punktestand 7,5 von 10.

Auch Camunda erzielt in beiden Kriterien jeweils 3 Punkte, da nativ Workitems einer Task automatisch ausgeführt werden können, aber keine vordefinierten Typen, wie in Axon Ivy, vorliegen. Allerdings können in Camunda unabhängig vom Kontext in jedem Task-Typ Skripte hinterlegt werden, mithilfe des Task Listeners und des „Output“-Tabs. Die Workitems von jedem Task-Typ außer User Task werden automatisch, ohne zugeordnete Ressource, ausgeführt. Die Task-Typen Receive, Send und Call sind ähnlich wie in Bonitasoft, außer dass Skripte in diesen freier hinterlegt werden können. Eine Service Task kann genutzt werden, um externe Systeme bzw. APIs anzusprechen und

eine Business Rule Task kann Regeln aus einer DMN-Entscheidungstabelle ausführen. Die Konnektoren zu externen Systemen müssen, wie bei Bonitasoft, vollkommen in Skripten konfiguriert werden. Die Workitems der Script Task führen individuelle Skripte aus und können somit sehr variabel eingesetzt werden.
Zusammengefasst erhält Camunda für das Pattern „Automatic Execution“ somit den Gesamtpunktestand 7,5 von 10.

| | Axon Ivy | Bonitasoft | Camunda | YAWL |
|---|---|---|---|---|
| **1. Direct Distribution** | 5 | 10 | 7,5 | 10 |
| **2. Role-Based Distribution** | 9,17 | 10 | 7,22 | 10 |
| **3. Deferred Distribution** | 8 | 4 | 8 | 8 |
| **4. Authorization** | 5 | 0 | 6,67 | 5 |
| **5. Separation of Duties** | 0 | 0 | 0 | 8 |
| **6. Case Handling** | 8,75 | 10 | 7,5 | 10 |
| **7. Retain Familiar** | 5 | 10 | 5 | 10 |
| **8. Capability-Based Distribution** | 4,29 | 4,29 | 0 | 7,86 |
| **9. History-Based Distribution** | 0 | 0 | 0 | 3,33 |
| **10. Organisational Distribution** | 7,5 | 9,17 | 0 | 10 |
| **11. Automatic Execution** | 10 | 7,5 | 7,5 | 7,5 |

Abbildung 7: Ergebnisse Creation Patterns

### 3.2.2 Untersuchung der Push Patterns

Für das Pattern „Distribution by Offer - Single Resource“, welches dem Zustandstransfer „S:offer_s“ entspricht [93], wurden die folgenden drei Kriterien untersucht:

- 1. Die Möglichkeit, ein Workitem an eine ausgewählte einzelne Ressource unverbindlich zuzuordnen
  - Gewichtung: *1
- 2. Das Workitem wird nur der angegebenen Ressource angezeigt
  - Gewichtung: *1
- 3. Das Workitem wird im Status „offered“ also nicht verpflichtend (oder äquivalentem Status) angeboten, sodass die Ressource das Workitem erst annehmen muss (claim oder äquivalente Aktion) bevor es ihr zugeteilt ist
  - Gewichtung: *0,5

Axon Ivy erhält für die ersten beiden Kriterien jeweils 4 Punkte, da die im Pattern „Direct Distribution“ beschriebene Möglichkeit Workitems einer Ressource zuzuordnen dies standardmäßig unverbindlich ausführt. Indem das „Responsible“-Feld einer User Task verwendet, und entsprechend ein Username oder eine Rolle mit nur einem User angegeben wird, werden sämtliche Workitems dieser Task automatisch nur der entsprechenden Ressource angeboten und angezeigt.
Auch für das 3. Kriterium erhält Axon Ivy 4 Punkte, da über die oben beschriebene Möglichkeit Workitems nativ unverbindlich in Axon Ivy angeboten werden und erst angenommen werden müssen, damit diese einer Ressource zugeteilt sind. Während das Workitem einer Ressource angeboten wird, befindet es sich im Status „OPEN (SUSPENDED)“. Um das Workitem anzunehmen, muss eine Ressource in der Weboberfläche den Button „Park task“ klicken. Dadurch wird das Workitem ihr zugeteilt und der Status wird in „OPEN (PARKED)“ geändert, wodurch das Workitem automatisch nur noch für diese Ressource sichtbar aber noch nicht gestartet ist. Dementsprechend entspricht „OPEN (SUSPENDED)“ in Axon Ivy dem „offered“-Status und „OPEN (PARKED)“ dem

„allocated“-Status.
Axon Ivy erzielt abschließend in dem Pattern „Distribution by Offer - Single Resource“ die Punktzahl 10 von 10.

Auch Bonitasoft erzielt in den ersten beiden Kriterien 4 Punkte, da die Workitems standardmäßig durch das Actor Mapping unverbindlich angeboten werden und nur für die angegebenen Ressourcen sichtbar sind. Um das aktuelle Pattern abzubilden, wird dem entsprechenden Actor nur eine einzelne Ressource zugewiesen.
Für das 3. Kriterium erhält Bonitasoft auch 4 Punkte, da sich dieses Kriterium mit dem nativen Vorgehen von Bonitasoft deckt. Die Workitems werden durch das Actor Mapping unverbindlich angeboten und befinden sich im Status „Ready“ und das Feld „Assigned to“ ist leer. Bevor ein Workitem einer Ressource zugeteilt ist, muss diese in der Weboberfläche den Button „take“ klicken, wodurch die Ressource im Feld „Assigned to“ des Workitems eingetragen wird. Der Name des Status ändert sich nicht. Nachdem ein Workitem durch den „take“-Button angenommen wurde, wird es nur noch dieser Ressource angezeigt und sie kann mit der Bearbeitung beginnen. Folglich entspricht in Bonitasoft der Status „Ready“ ohne eine eingetragene Ressource bei „Assigned to“ dem „offered“-Status und der Status „Ready“ mit einer eingetragenen Ressource bei „Assigned to“ dem „allocated“-Status.
Zusammengefasst erzielt Bonitasoft in dem Pattern „Distribution by Offer - Single Resource“ die Punktzahl 10 von 10.

Für die ersten beiden Kriterien erhält Camunda jeweils 4 Punkte, da die Workitems nativ unverbindlich angeboten werden und nur für die angegebenen Ressourcen sichtbar sind, wenn der Name der Ressource bei dem „Candidate Users“-Feld eingetragen ist. Falls nur ein Username in dem Feld „Candidate Users“ eingetragen wird, ist das aktuelle Pattern dadurch umgesetzt.
Camunda erzielt auch im 3. Kriterium 4 Punkte, da durch die Verwendung von dem Feld „Candidate Users“ die Workitems nativ im Status „UNASSIGNED“ angeboten werden und die entsprechende Ressource diese erst in der Weboberfläche mit einem Klick auf den „Claim“-Button annehmen muss, um damit zu interagieren. Nachdem die Ressource das Workitem mithilfe des „Claim“-Buttons angenommen hat, ändert sich der Status zu „ASSIGNED“, wodurch nur noch die entsprechende Ressource mit dem Workitem interagieren kann. Somit entspricht in Camunda der Status „UNASSIGNED“ dem „offered“-Status und der Status „ASSIGNED“ dem „allocated“-Status. Der Status eines Workitems kann in Camunda nicht eingesehen werden, weder in den Weboberflächen noch in einem anderen Fenster oder per API. Somit können die Status aus dem Task Lifecycle [52] in dem Tool nicht wiedergefunden werden. Daher kann der Status eines Workitems nur vermutet werden, indem er aus dem Task Lifecycle [52] abgeleitet wird. Hierfür werden die Funktionen, die für ein Workitem zu einem bestimmten Zeitpunkt im Tool möglich sind, mit den Operations aus dem Task Lifecycle [52] abgeglichen. Auf diese Weise können die oben beschriebenen Vermutungen über den Status aufgestellt werden. Da die Funktionen im Tool zu den entsprechenden Zeitpunkten mit den Operations aus dem Task Lifecycle komplett übereinstimmen, sind die Vermutungen äußerst wahrscheinlich. Daher werden diese Vermutungen über die Status für die weiteren Patterns als gegeben angenommen und Camunda kann somit auch in Kriterien, welche sich explizit auf den Status beziehen die volle Punktzahl erreichen. Die Aspekte, welche die Kriterien erfüllen, die unabhängig von dem Namen eines Status sind, können im Tool nachvollzogen werden und sind somit gültig. Dies gilt auch für alle folgenden Patterns.
Camunda erhält somit für das Pattern „Distribution by Offer - Single Resource“ die Gesamtpunkzahl 10 von 10.

Der Zustandstransfer „S:offer_m“ wird durch das Pattern „Distribution by Offer - Multiple Resources“ abgebildet [93]. Für das Pattern wurden die folgenden drei Kriterien betrachtet:

- 1. Die Möglichkeit, ein Workitem einer Gruppe ausgewählter Ressourcen zuzuordnen auf unverbindlicher Basis

    - Gewichtung: *1
- 2. Das Workitem wird allen Ressourcen im „offered“-Status also nicht verpflichtend (oder äquivalentem Status) angeboten, sodass eine Ressource das Workitem erst annehmen muss (claim oder äquivalente Aktion), bevor es ihr zugeteilt ist
    - Gewichtung: *0,5
- 3. Die anderen Ressourcen können das zugeteilte Workitem nicht mehr auswählen bzw. damit interagieren
    - Gewichtung: *1

Für die ersten beiden Kriterien erhält Axon Ivy jeweils 4 Punkte, da die im Pattern „Role-Based Distribution“ beschriebene Möglichkeit, Workitems durch Rollen standardmäßig unverbindlich an Ressourcen zuweist. Die gewünschten User werden einer Rolle hinzugefügt und diese wird im „Responsible“-Feld der User Task hinterlegt. Zur Laufzeit werden sämtliche Workitems der Task allen Usern, welche der Rolle zugewiesen sind, unverbindlich angeboten. Die Workitems sind nur für diese User sichtbar. Wie in „Distribution by Offer - Single Resource“ beschrieben, sind die angebotenen Workitems im Status „OPEN (SUSPENDED)“. Damit ein Workitem einem User fest zugeteilt ist, muss dieser das Workitem durch den „Park task“-Button erst annehmen, wodurch sich auch der Status zu „OPEN (PARKED)“ ändert.
In dem 3. Kriterium erzielt Axon Ivy auch 4 Punkte, da das Workitem, nachdem es angenommen wurde, automatisch nur noch dem User angezeigt wird, der es allokiert hat. Es wird nativ aus den Listen der restlichen User der Rolle entfernt, sodass sie das Workitem dann weder sehen noch damit interagieren können.
Zusammenfassend erzielt Axon Ivy in dem Pattern „Distribution by Offer - Multiple Resources“ die Punktzahl 10 von 10. Allerdings muss angemerkt werden, dass es in Axon Ivy nicht möglich ist, ein Workitem mehreren Ressourcen einzeln oder mehreren unabhängigen Rollen anzubieten.

Bonitasoft erhält in allen drei Kriterien jeweils 4 Punkte, da diese nativ durch das Actor Mapping umgesetzt sind. Mithilfe des Actor Mappings kann, wie in „Role-Based Distribution“ beschrieben, einem Actor eine oder mehrere Rollen zugewiesen werden. Die Workitems der Task, welche den entsprechenden Actor beinhaltet, werden jedem User unverbindlich angeboten, der mindestens eine der zugewiesenen Rollen besitzt. Wie in „Distribution by Offer - Single Resource“ beschrieben, muss ein User ein Workitem zuerst mithilfe des „take“-Buttons annehmen, bevor dieses ihm zugeteilt ist. Nachdem ein User ein Workitem angenommen hat, kann nur noch dieser damit interagieren, da es automatisch aus den Listen der anderen potenziellen User entfernt wird.
Dementsprechend erhält Bonitasoft für das Pattern „Distribution by Offer - Multiple Resources“ die Gesamtpunktzahl 10 von 10. In Bonitasoft kann ein Workitem auch mehreren Ressourcen einzeln oder mehreren unabhängigen Rollen angeboten werden, indem diese auf denselben Actor zugewiesen werden.

Camunda erzielt für das 1. und 2. Kriterium jeweils 4 Punkte, da diese Kriterien, wie in „Role-Based Distribution“ beschrieben, durch das „Candidate Groups“-Feld nativ abgebildet sind. Im Feld „Candidate Groups“, können beliebig viele Gruppennamen hinterlegt werden, sodass ein Workitem dieser Task allen Usern unverbindlich angeboten wird, welche mindestens einer dieser Gruppen angehören. Nach dem gleichen Prinzip wie in „Distribution by Offer - Single Resource“, wird das Workitem nativ im Status „UNASSIGNED“ angeboten und eine der potenziellen Ressourcen kann dieses durch den „Claim“-Button annehmen. Danach ändert sich der Status zu „ASSIGNED“ und die Ressource kann mit dem Workitem interagieren.
Für das 3. Kriterium erhält Camunda allerdings nur 2 Punkte, da Code verwendet werden muss für eine Berechtigung, um dieses Kriterium zu erfüllen. Ohne diesen Code, wird ein Workitem, welches wie oben beschrieben angenommen wurde, weiterhin den anderen potenziellen Usern angezeigt und diese können damit interagieren. Sie könnten den zugehörigen User ändern und das Workitem sich selbst oder einem anderen User zuordnen oder es abschließen. Es ist möglich, in Camunda nativ die Berechtigung der

Gruppe auf das Workitem zu entfernen, wodurch nur noch der zugeteilte User das Workitem sieht. Allerdings muss dies manuell erfolgen, was in einem produktiven System nicht möglich bzw. sinnvoll ist. Außerdem wird auf diese Weise das Workitem den restlichen Usern der Gruppe nicht mehr angezeigt, falls der zugeteilte User es wieder freigibt, da die Berechtigung entfernt wurde. Somit ist dieser Ansatz nicht zielführend. Daher muss die Standardberechtigung der User für angebotene und zugeteilte Workitems in Camunda per Code geändert werden, um das Kriterium umzusetzen. Diese wird von „TASK_UPDATE“ auf „TASK_WORK“ geändert. Auf diese Weise können die restlichen potenziellen User das Workitem zwar noch sehen, aber nicht mehr damit interagieren. Mit dem Workitem integrieren kann nur der zugeteilte User. Die volle Codezeile, die für den Zweck in der „default.yml“ ergänzt werden muss, lautet: „generic-properties: properties: default-user-permission-name-for-task: TASK_WORK“.
Insgesamt erzielt Camunda somit die Punktzahl 8 von 10 für das Pattern „Distribution by Offer - Multiple Resources“. Außerdem kann in Camunda ein Workitem auch mehreren Ressourcen einzeln oder mehreren unabhängigen Rollen bzw. hier Gruppen angeboten werden, indem diese in dem Feld „Candidate Users“ bzw. „Candidate Groups“ hinterlegt werden.

Für das nächste Pattern „Distribution by Allocation - Single Resource“, welches dem Zustandstransfer „S:allocate“ entspricht [93], wurden zwei Bewertungskriterien untersucht:

- 1. Die Möglichkeit, ein Workitem einer bestimmten Ressource zur Ausführung zuzuordnen auf einer verbindlichen Basis
    - Gewichtung: *1
- 2. Das Workitem wird direkt einer Ressource zugeteilt im Status „allocated“, ohne es vorher anzubieten (es muss nicht erst angenommen werden)
    - Gewichtung: *0,5

Axon Ivy erzielt für beide Kriterien jeweils 1 Punkt, da diese nur durch zwei Zeilen Code umsetzbar sind und der gewünschte User automatisch eingeloggt werden muss. Nach den Erkenntnissen aus den beiden „Distribution by Offer“-Patterns muss ein Workitem einem User im Status „OPEN (PARKED)“ zugeteilt werden, damit dies verbindlich ist. Da der Standardweg im Tool allerdings die Workitems immer nur anbietet und nicht fest zuteilt, kann dies nur selbst programmiert werden. Hierfür muss der Befehl „ivy.session.parkTask(task)“ verwendet werden. Allerdings teilt dieser das Workitem immer nur dem aktuellen Session-User zu, also dem User, der gerade eingeloggt ist. Daher muss auch der Session-User geändert werden mit dem Befehl „ivy.session.loginSessionUser( username, passwort);“. Zusammengesetzt müssen diese Befehle im „Code“-Abschnitt des „Task“-Tabs einer User Task hinterlegt werden. Auf diese Weise wird ein Workitem dieser Task automatisch dem angegebenen User verbindlich zugeteilt und nicht nur angeboten. Allerdings wird automatisch der hinterlegte User in die aktuelle Session eingeloggt, was ggf. auch nicht gewollt und/oder möglich ist in einem produktiven System. In Bonitasoft muss auch Code verwendet werden, aber dort gibt es nicht das Problem, dass der User automatisch eingeloggt wird und der vorherige User ausgeloggt. Daher erhält Axon Ivy hier weniger Punkte.
Abschließend erhält Axon Ivy für das Pattern „Distribution by Allocation - Single Resource“ die Punktzahl 2,5 von 10.

Für die beiden Bewertungskriterien erhält Bonitasoft jeweils 2 Punkte, da zwei Zeilen Code erforderlich sind, um die Kriterien umzusetzen. Wie in den beiden „Distribution by Offer“-Patterns erläutert, müssen die Workitems im Status „Ready“ mit einer eingetragenen Ressource bei „Assigned to“, welches dem „allocated“-Status entspricht, zugeteilt werden, damit dies verbindlich ist. Durch das Actor Mapping werden die Workitems den Usern nur angeboten. Für eine verbindliche Zuteilung muss ein Actor Filter verwendet werden. Dieser kann in jeder Swimlane oder Human Task im „Actor Filter“-Feld hinterlegt werden. Durch einen Actor Filter wird ein Workitem direkt einem User zugeteilt, ohne dass dieser das Workitem annehmen muss (s. digitaler Anhang – Öffentlich). Der Actor

Filter „Single user“ kann dafür verwendet werden. Allerdings müssen im Input des Filters zwei Zeilen Code verwendet werden, da der Filter die ID des Users und nicht dessen Namen als Input benötigt (s. digitaler Anhang - Beantwortet). Die ID lässt sich nur durch Code erhalten. Demnach müssen die folgenden zwei Zeilen Code hinterlegt werden (s. digitaler Anhang - Beantwortet): „def user = apiAccessor.getIdentityAPI().getUserByUserName(„Sue“);“ und „return user.getId();“. Der erste Befehl kann in Bonitasoft durch eine GUI auf Knopfdruck generiert werden, sodass nur der zweite Befehl eigenständig programmiert werden muss. Sowohl der Actor Filter als auch die GUI zur Generierung des ersten Befehls werden standardmäßig im Tool zur Verfügung gestellt, da allerdings Code verwendet werden muss, kann trotzdem nur die entsprechende Punktzahl vergeben werden.
Zusammengefasst erzielt Bonitasoft somit die Punktzahl 5 von 10 für das Pattern „Distribution by Allocation - Single Resource“.

Camunda erhält in beiden Kriterien jeweils 4 Punkte, da diese nativ über die „Assignee“-Funktionalität abbildbar sind. Wie in den beiden „Distribution by Offer“-Patterns erläutert, müssen die Workitems im Status „ASSIGNED“ zugeordnet werden, damit sie verbindlichen zugeteilt sind. Hierfür kann in Camunda standardmäßig der Usernamen im Feld „Assignee“ in einer User Task als String hinterlegt werden. Diesem User werden dann sämtliche Workitems der Task im Status „ASSIGNED“ zugeteilt, sodass dieser die Workitems direkt bearbeiten kann und nicht erst annehmen muss.
Camunda erhält somit den finalen Punktestand 10 von 10 für das Pattern „Distribution by Allocation - Single Resource“. Zusätzlich ist anzumerken, dass durch die Genehmigung aus „Distribution by Offer - Multiple Resources“ zusätzlich sichergestellt wird, dass der User das Workitem nicht mehr abgeben kann, wodurch es komplett verbindlich zugeteilt ist.

Für das Pattern „Random Allocation“ wurde folgendes Kriterium untersucht:

- 1. Die Möglichkeit, ein Workitem einer ausgewählten Ressource zuzuordnen, die nach dem Zufallsprinzip aus einer Gruppe in Frage kommender Ressourcen ausgewählt wird
    - Gewichtung: *1

In Axon Ivy kann das Kriterium nur umgesetzt werden, indem die Logik mithilfe von zur Verfügung gestellten APIs programmiert wird (s. digitaler Anhang - Beantwortet). Dementsprechend erhält Axon Ivy für das Kriterium 0 Punkte. Keine der erläuterten nativen Zuordnungsfunktionalitäten verfügt über die Möglichkeit, einen zufälligen User aus einer Gruppe zu wählen (s. digitaler Anhang - Beantwortet).
Für das Pattern „Random Allocation“ erhält Axon Ivy somit die Gesamtpunktzahl 0 von 10.

Bonitasoft erzielt in dem Kriterium ebenfalls 0 Punkte, da es nur durch individuell programmierte Logik umgesetzt werden kann. Damit ein Workitem dynamisch zugeordnet wird, muss ein Actor Filter verwendet werden. Keiner der Standard Actor Filter verfügt über eine entsprechende Funktionalität [25]. Demnach muss ein benutzerdefinierter Actor Filter individuell programmiert werden, der die zufällige Auswahl eines Users abbildet. Hierbei könnte die API verwendet werden, um alle potenziellen User für eine Task zu finden (s. digitaler Anhang - Öffentlich).
Insgesamt erzielt Bonitasoft in dem Pattern „Random Allocation“ die Punktzahl 0 von 10.

Auch Camunda erzielt im Kriterium 0 Punkte, da das Kriterium nur durch programmierte Logik mithilfe einer API möglich ist. Die Zuteilung von Workitems erfolgt durch die Angabe eines Usernamens im „Assignee“-Feld der entsprechenden User Task. Damit dieser User zufällig variabel gewählt werden kann, muss eine Variable hinterlegt werden, die zur Laufzeit belegt wird. Dies ist nur durch das Programmieren der gewünschten Logik möglich. Hier könnte eine REST-API unterstützen, um alle potenziellen User der im „Candidate Groups“-Feld eingetragenen Rolle zu finden (s. digitaler Anhang - Öffentlich).

Abschließend erhält Camunda für das Pattern „Random Allocation“ somit die Punktzahl 0 von 10.

Bei dem Pattern „Round Robin Allocation“ wurden folgende zwei Kriterien betrachtet:

- 1. Die Möglichkeit, ein Workitem einer ausgewählten Ressource zuzuordnen, die aus einer Gruppe von in Frage kommenden Ressourcen auf zyklischer Basis ausgewählt wird
    - Gewichtung: *1
- 2. Es ist möglich, entweder die Ressource zu wählen, die ein Workitem dieser Task am längsten nicht gemacht hat oder die Ressource zu nehmen, die am wenigsten Workitems dieser Task ausgeführt hat (dafür muss die Anzahl gespeichert werden)
    - Gewichtung: *0,5

Beide Kriterien erhalten in Axon Ivy jeweils 0 Punkte, da diese nur umgesetzt werden können, indem die Logik programmiert wird (s. digitaler Anhang - Beantwortet). Hierbei können vorgefertigte APIs unterstützen, um z.B. alle potenziellen User einer Rolle zu finden (s. digitaler Anhang - Öffentlich). Es liegt keine Möglichkeit vor, das Verhalten nativ in Axon Ivy abbilden zu können (s. digitaler Anhang - Beantwortet).
Zusammenfassend erhält Axon Ivy für das Pattern „Round Robin Allocation“ die Punktzahl 0 von 10.

Bonitasoft erhält für die beiden Kriterien 0 Punkte. Wie in „Random Allocation“ beschrieben, liegt keine native Möglichkeit vor, das gewünschte Verhalten der beiden Kriterien abzubilden. Die einzige Möglichkeit ist, die gesamte Logik selbst zu programmieren, wobei die API aus „Random Allocation“ unterstützen könnte. Auch auf Nachfrage konnte keine Antwort gefunden werden (s. digitaler Anhang - Unbeantwortet).
Insgesamt erzielt Bonitasoft die Punktzahl 0 von 10 für das Pattern „Round Robin Allocation“.

Für die beiden Kriterien erhält Camunda jeweils 0 Punkte, da diese nur umgesetzt werden können, indem die Logik mithilfe von APIs programmiert wird (s. digitaler Anhang - Öffentlich). Aus den gleichen Gründen wie bei „Random Allocation“ muss die Logik programmiert werden. Zusätzlich zur beschriebenen REST-API, liegt die API vor, um die Ausführungshistorie einer Task zu erhalten, wie in „History-Based Distribution“ beschrieben.
Abschließend erhält Camunda für das Pattern „Round Robin Allocation“ die Punktzahl 0 von 10.

Für das Pattern „Shortest Queue“ wurde das folgende Kriterium untersucht:

- 1. Die Möglichkeit, ein Workitem einer ausgewählten Ressource zuzuordnen, die aus einer Gruppe in Frage kommender Ressourcen auf der Grundlage des kürzesten Arbeitsvorrates ausgewählt wurde
    - Gewichtung: *1

Für das Kriterium erhält Axon Ivy 0 Punkte, da es sich nur umsetzen lässt, indem es mit einer Task-Query programmiert wird (s. digitaler Anhang - Beantwortet). Die Task-Query liefert die Anzahl der Workitems, an denen ein User arbeiten kann. Den Import für die Task-Query stellt Axon Ivy zur Verfügung, alles andere muss individuell programmiert werden (s. digitaler Anhang - Beantwortet).
Insgesamt erhält Axon Ivy für das Pattern „Shortest Queue“ somit die Punktzahl 0 von 10.

Bonitasoft kann das Kriterium nicht abbilden und erhält daher 0 Punkte. Auch auf Nachfrage konnte keine Lösung gefunden werden (s. digitaler Anhang - Unbeantwortet). Es liegt kein passender Actor Filter vor und die entsprechende Logik müsste eigenständig programmiert werden.
Bonitasoft erzielt dementsprechend bei dem Pattern „Shortest Queue“ die Punktzahl 0 von 10.

In Camunda lässt sich das Kriterium nur durch individuell programmierte Logik umsetzen, wobei unterstützende APIs zur Verfügung gestellt werden (s. digitaler Anhang - Öffentlich). Daher erhält Camunda für das Kriterium 0 Punkte. Die APIs, um die potenziellen User für eine Task zu finden und um herauszufinden, wie viele Workitems einem bestimmten User zugeteilt sind (s. digitaler Anhang - Öffentlich), könnten bei der Umsetzung verwendet werden.
Zusammengefasst erzielt Camunda die Punktzahl 0 von 10 für das Pattern „Shortest Queue“.

Für das nächste Pattern „Early Distribution“ wurden die folgenden vier Kriterien betrachtet:

- 1.1 Die Möglichkeit, ein Workitem anzukündigen, bevor es tatsächlich aktiviert wird
    - Gewichtung: *1
- 1.2 Die Möglichkeit, ein Workitem möglicherweise an Ressourcen zuzuordnen, bevor es tatsächlich aktiviert wird
    - Gewichtung: *0,5
- 2. Die Ankündigung/Zuordnung erfolgt direkt zu Prozessstart (zu Beginn des Cases)
    - Gewichtung: *0,5
- 3. Das früh zugeteilte Workitem kann erst ausgeführt werden, wenn der Prozess an der entsprechenden Stelle bzw. Task ankommt
    - Gewichtung: *1

Axon Ivy erhält für alle vier Kriterien 0 Punkte, da ein Workitem einer Task erst erstellt wird, wenn der Prozess bei der Task angekommen ist (s. digitaler Anhang - Beantwortet). Dementsprechend kann ein Workitem vorher bzw. zu Prozessstart weder angekündigt noch zugeteilt werden, da noch nichts existiert, auf das verwiesen werden könnte (s. digitaler Anhang - Beantwortet). Das 3. Kriterium ist daher auch nicht möglich, da es keine Möglichkeit gibt, Workitems früher zuzuteilen.
Axon Ivy erhält final für das Pattern „Early Distribution“ den Punktestand 0 von 10.

Auch Bonitasoft verfügt über keine Möglichkeit, die vier Kriterien umzusetzen und erzielt somit in jedem Kriterium 0 Punkte. Auch auf Nachfrage wurde keine Antwort gefunden (s. digitaler Anhang - Unbeantwortet).
Zusammengefasst erhält Bonitasoft den Punktestand 0 von 10 für das Pattern „Early Distribution“.

Alle vier Kriterien erhalten in Camunda 0 Punkte, da diese nicht in Camunda abgebildet werden können. Eine Ankündigung muss individuell per Task Listener programmiert werden, allerdings wird der Code erst ausgeführt, wenn der Prozess an der Task angekommen ist (s. digitaler Anhang - Beantwortet). Dementsprechend kann ein Workitem vorher bzw. zu Prozessstart weder angekündigt noch zugeteilt werden, da der Prozess dafür erst bei der entsprechenden Task ankommen muss (s. digitaler Anhang - Beantwortet). Auch das 3. Kriterium kann nicht umgesetzt werden, da es kein früher zugeteiltes Workitem gibt. Auf eine erneute Nachfrage zur Klarstellung folgte keine Antwort, daher ist davon auszugehen, dass das Workitem erst existiert, wenn der Prozess bei der Task angekommen ist (s. digitaler Anhang - Beantwortet).
Camunda erzielt somit den Punktestand 0 von 10 für das Pattern „Early Distribution“.

Bei dem Pattern „Distribution on Enablement“ wurden die folgenden drei Kriterien betrachtet:

- 1. Die Möglichkeit, ein Workitem in dem Moment auszuschreiben und Ressourcen zuzuordnen, wenn die Task, der das Workitem entspricht, zur Ausführung freigegeben wird
    - Gewichtung: *1
- 2. Das Workitem kann indirekt zugeordnet werden also angeboten werden
    - Gewichtung: *0,5

- 3. Das Workitem kann direkt zugeordnet werden also zugeteilt werden
    - Gewichtung: *0,5

Axon Ivy erzielt in dem 1. Kriterium 4 Punkte, da dies die Standardfunktionalität ist, wie im Tool Workitems zugeordnet werden. Wie im Pattern „Early Distribution“ erläutert, wird ein Workitem erst zu dem Zeitpunkt erstellt, an dem der Prozess an der entsprechenden Task angelangt ist. Dementsprechend wird das Workitem auch zu diesem Zeitpunkt den Usern zugeordnet.
Auch für das 2. Kriterium erhält Axon Ivy 4 Punkte, da Workitems, wie in den beiden „Distribution by Offer“-Pattern erklärt, den Usern bei der Zuordnung standardmäßig angeboten werden. Hierbei ist es sowohl möglich, das Workitem einem User als auch mehreren Usern anzubieten.
Bei dem 3. Kriterium erzielt Axon Ivy 1 Punkt, da dieses Verhalten, wie in „Distribution by Allocation - Single Resource“ erläutert, nur durch zwei Zeilen Code umsetzbar ist. Aus den gleichen Gründen wie bei „Distribution by Allocation - Single Resource“ wird auch hier nur 1 Punkt vergeben.
Zusammengerechnet erhält Axon Ivy für das Pattern „Distribution on Enablement“ somit den Punktestand 8,13 von 10.

Bonitasoft erhält für das 1. Kriterium 4 Punkte, da auch hier das Kriterium durch die Standardfunktionalität des Tools zur Zuordnung von Workitems abgebildet wird. Sobald der Prozess an der entsprechenden Task angekommen ist, wird das Workitem erzeugt und den entsprechenden Usern zugeordnet.
Auch in dem 2. Kriterium erzielt Bonitasoft 4 Punkte, da Workitems, wie in den „Distribution by Offer“-Patterns beschrieben, durch das Actor Mapping den Usern standardmäßig angeboten werden. Hierbei ist es möglich, das Workitem sowohl einem User als auch mehreren Usern anzubieten.
Für das 3. Kriterium erhält Bonitasoft 2 Punkte, da dies, wie in „Distribution by Allocation - Single Resource“ beschrieben, nativ mit einem Actor Filter möglich ist, aber dieser zwei Zeilen Code benötigt.
Abschließend erhält Bonitasoft daher die Punktzahl 8,75 von 10 für das Pattern „Distribution on Enablement“.

Bei dem 1. Kriterium erzielt Camunda 4 Punkte, da auch hier das Kriterium durch die Standardfunktionalität des Tools zur Zuordnung von Workitems abgebildet wird. Sobald der Prozess an der entsprechenden Task angekommen ist, wird das Workitem erzeugt und den entsprechenden Usern zugeordnet, was sich aus der Antwort für das Pattern „Early Distribution“ ableiten lässt.
Für das 2. und 3. Kriterium erhält Camunda jeweils 4 Punkte, da sich diese Kriterien, wie in den beiden „Distribution by Offer“-Patterns und dem „Distribution by Allocation“-Pattern erläutert, nativ im Tool umsetzen lassen durch die Felder „Candidate Groups“ und „Candidate Users“ für das Anbieten und „Assignee“ für das Zuteilen.
Insgesamt erzielt Camunda damit für das Pattern „Distribution on Enablement“ die Punktzahl 10 von 10.

Für das Pattern „Late Distribution“ wurden die folgenden drei Bewertungskriterien betrachtet:

- 1. Die Möglichkeit, ein Workitem auszuschreiben und Ressourcen zuzuordnen, nachdem die Task, zu der das Workitem gehört, für die Ausführung freigegeben worden ist
    - Gewichtung: *1
- 2. Es wird gewartet mit der Zuordnung an einen User, bis inhaltliche Faktoren (Anzahl der Workitems auf der Worklist, verfügbare Ressourcen, ...) erfüllt sind
    - Gewichtung: *0,5
- 3. Es wird gewartet mit der Zuordnung an einen User, bis eine Zeit abgelaufen ist
    - Gewichtung: *0,5

In allen drei Tools kann das Verhalten des Patterns mithilfe von BPMN-Elementen (Exclusive Gateways und Timer Events) abgebildet werden, obwohl [93] besagt, dass BPMN dieses Pattern nicht unterstützt.

Axon Ivy erhält für das 1. Kriterium 4 Punkte, da es möglich ist, mit der Zuordnung des Workitems sowohl zu warten, bis eine Zeit abgelaufen ist, als auch bis eine Bedingung erfüllt ist. Beides ist standardmäßig im Tool mithilfe von BPMN-Elementen möglich. Dies wird für die nächsten beiden Kriterien genauer erläutert.
Für das 2. Kriterium erhält Axon Ivy 2 Punkte, da es möglich ist zu warten, bis eine Bedingung erfüllt ist, aber für die Überprüfung bzw. Definition der Bedingung immer Code verwendet werden muss. Vor die Task, welche verzögert werden soll, kann ein Exclusive Gateway modelliert werden, welches eine Variable nach einem Wert überprüft. Die Überprüfungsoperation für die Variable und die Eingabe des gewünschten Wertes können nur per Code erfolgen. Sollte dieser Wert erfüllt sein, kann der Pfad zur entsprechenden Task fortgeführt werden. Ansonsten wird ein anderer Pfad verfolgt, der die Bedingung nach einer vorgegebenen Zeit erneut überprüft oder eine entsprechende Aufgabe ausführt, nach der die Bedingung erneut geprüft werden kann. Auf diese Weise werden die Workitems der gewünschten Task erst zugeordnet, nachdem die Bedingung erfüllt ist.
Axon Ivy erzielt im 3. Kriterium 4 Punkte, da nativ in einer User Task im „Task“-Tab unter „Delay“ eine Wartezeit hinterlegt werden kann. Erst nachdem diese Wartezeit abgelaufen ist, wird das Workitem der Task einem User angeboten oder zugeteilt. Die Wartezeit muss als „Duration“ in dem Format des folgenden Beispiels hinterlegt werden: „0H0M5S“.
Insgesamt erhält Axon Ivy somit für das Pattern „Late Distribution“ die Punktzahl 8,75 von 10.

Auch Bonitasoft erhält für das 1. Kriterium 4 Punkte, da die Zuordnung eines Workitems sowohl verzögert werden kann aufgrund einer Zeit als auch aufgrund einer Bedingung. Beide Möglichkeiten können nativ durch BPMN-Elemente umgesetzt werden. Dies wird für die nächsten beiden Kriterien genauer erklärt.
Für das 2. Kriterium erhält Bonitasoft 2 Punkte, da das Abwarten, bis eine Bedingung erfüllt ist, standardmäßig mit einem Exclusive Gateway möglich ist, aber für die Überprüfung bzw. Definition der Bedingung immer Code verwendet werden muss. Das Vorgehen ist dabei deckungsgleich mit dem, welches für Axon Ivy erläutert wurde, da die gleichen BPMN-Elemente verwendet werden.
Bonitasoft erhält im 3. Kriterium 4 Punkte, da die Zuteilung eines Workitems nativ durch ein vorgeschobenes Timer Event verzögert werden kann. Hier kann sowohl eingestellt werden, ob bis zu einem bestimmten Datum samt Uhrzeit gewartet werden soll oder ob ein gewisses Zeitintervall abgewartet werden soll. Beides ist durch GUI-Elemente einstellbar und auf Knopfdruck wird der passende Code zur Eingabe erzeugt und hinterlegt.
Abschließend erhält Bonitasoft daher den Punktestand 8,75 von 10 für das Pattern „Late Distribution“.

Camunda erzielt im 1. Kriterium 4 Punkte, da auch hier, wie bei den anderen Tools, die Zuordnung eines Workitems sowohl verzögert werden kann aufgrund einer Zeit als auch aufgrund einer Bedingung mithilfe von BPMN-Elementen.
Für das 2. Kriterium erhält Camunda 2 Punkte. Das Verhalten kann nativ durch ein Exclusive Gateway oder ein Conditional Event umgesetzt werden. Allerdings muss für die Überprüfung bzw. Definition der Bedingung immer Code verwendet werden. Das Vorgehen ist auch hier das gleiche wie bei Axon Ivy, nur dass statt einem Exclusive Gateway auch ein Conditional Event eingefügt werden kann, welches dann die gewünschte Bedingung beinhaltet und überprüft.
In dem 3. Kriterium erzielt Camunda 4 Punkte, da das Verhalten standardmäßig durch ein Timer Event abgebildet werden kann. Bei diesem kann ausgewählt werden, ob bis zu einem bestimmten Datum abgewartet werden soll oder, ob ein Zeitintervall abgewartet werden soll oder, ob in einem bestimmten Zyklus etwas ausgelöst werden soll. Die entsprechenden Formate sind mit Beispielen direkt im Tool hinterlegt. Auch ist zu

beachten, dass das Timer Event nicht an eine Task modelliert werden darf, da es sonst die Funktion ändert. Sollte das Timer Event an eine Task modelliert werden, laufen die entsprechenden Workitems dieser Task nach der angegebenen Zeit ab und sind damit beendet. Damit das Timer Event als Verzögerung fungiert, muss es unabhängig vor der gewünschten Task modelliert werden.
Zusammengefasst erhält Camunda somit bei dem Pattern „Late Distribution“ die Punktzahl 8,75 von 10.

| | Axon Ivy | Bonitasoft | Camunda | YAWL |
|---|---|---|---|---|
| **12. Distribution by Offer - Single Resource** | 10 | 10 | 10 | 10 |
| **13. Distribution by Offer - Multiple Resources** | 10 | 10 | 8 | 10 |
| **14. Distribution by Allocation - Single Resource** | 2,5 | 5 | 10 | 10 |
| **15. Random Allocation** | 0 | 0 | 0 | 10 |
| **16. Round Robin Allocation** | 0 | 0 | 0 | 9,17 |
| **17. Shortest Queue** | 0 | 0 | 0 | 10 |
| **18. Early Distribution** | 0 | 0 | 0 | 0 |
| **19. Distribution on Enablement** | 8,13 | 8,75 | 10 | 10 |
| **20. Late Distribution** | 8,75 | 8,75 | 8,75 | 8,75 |

Abbildung 8: Ergebnisse Push Patterns

### 3.2.3 Untersuchung der Pull Patterns

Das Pattern „Resource-Initiated Allocation“ entspricht den Zustandstransfers „R:allocate_s“ und „R:allocate_m“, je nachdem wie vielen Ressourcen das Workitem vorher angeboten wurde [93]. Für das Pattern wurden die folgenden drei Kriterien betrachtet:

- 1. Die Möglichkeit für eine Ressource, sich zu verpflichten, ein Workitem zu übernehmen, ohne dass sie sofort mit der Arbeit daran beginnen muss. Nach der Zuordnung ist die Ressource für die Ausführung des Workitems verantwortlich
  - Gewichtung: *1
- 2. Der Status des Workitem wird von „offered“ (oder äquivalenten Status) zu „allocated“ (oder äquivalenten Status) geändert
  - Gewichtung: *0,5
- 3. Es wird sichergestellt, dass nur die Ressource, die ein Workitem angenommen hat, auch daran arbeiten kann
  - Gewichtung: *0,5

Axon Ivy erzielt in allen drei Kriterien 4 Punkte, da User standardmäßig ein Workitem annehmen können, ohne dass mit der Arbeit daran direkt gestartet werden muss. Mithilfe des „park“-Buttons in der Weboberfläche, kann ein User das Workitem annehmen, sodass es nur noch ihm zugeteilt ist. Dadurch wird, wie im Pattern „Distribution by Offer - Single Resource“ beschrieben, der Status des Workitems von „OPEN (SUSPENDED)“ auf „OPEN (PARKED)“ geändert, was das 2. Kriterium erfüllt. Nachdem ein Workitem auf diese Weise angenommen wurde, wird es aus den Listen der anderen potenziellen User entfernt, sodass nur noch der zugeteilte User daran arbeiten kann.
Zusammengefasst erhält Axon Ivy für das Pattern „Resource-Initiated Allocation“ demnach den Punktestand 10 von 10.

Auch Bonitasoft erhält für alle drei Kriterien 4 Punkte. Nach der in „Distribution by Offer - Single Resource“ beschriebenen Vorgehensweise können Workitems angenommen werden. Dabei wird der Status, wie im 2. Kriterium gefordert, geändert. Außerdem wird

das Workitem automatisch aus den Listen der anderen potenziellen User entfernt, sobald ein User das Workitem angenommen hat. Auf diese Weise kann nur noch der zugeteilte User das Workitem bearbeiten. Nach dem Vorgehen aus „Distribution by Offer - Single Resource“ muss die Arbeit am Workitem auch nicht sofort begonnen werden. Es liegt kein extra Status und keine ausgewiesene Funktionalität vor, um ein Workitem zu starten. Falls ein User ein Workitem annimmt und danach diesem zugeteilt ist, kann dieser theoretisch direkt daran arbeiten. Dies wird auch in vielen folgenden Pattern als starten des Workitems interpretiert, allerdings werden hier trotzdem die vollen Punkte vergeben, da nur weil das Workitem gestartet wurde, dies nicht heißt, dass die Arbeit daran sofort erfolgen muss. Aufgrund des Prinzips, welches in „Selection Autonomy“ genauer beschrieben wird, kann ein Workitem einfach geschlossen werden und es kann an einem beliebigen anderen oder an keinem anderen Workitem weitergearbeitet werden. Demnach ist das Workitem durch das Zuteilen ggf. gestartet, allerdings muss die Arbeit daran nicht sofort beginnen. Es kann geschlossen und zu einem beliebigen anderen Zeitpunkt abgearbeitet werden.
Insgesamt erhält Bonitasoft somit für das Pattern „Resource-Initiated Allocation“ die Punktzahl 10 von 10.

In Camunda werden das 1. und das 2. Kriterium jeweils mit 4 Punkten bewertet, da, wie in „Distribution by Offer - Single Resource“ beschrieben, Workitems angenommen werden können und sich dabei der Status entsprechend des 2. Kriteriums ändert. Auch in Camunda liegt kein extra Status und keine ausgewiesene Funktionalität vor, um ein Workitem zu starten. Hier ist das gleiche Prinzip wie bei Bonitasoft auch auf Camunda anwendbar, wodurch es trotzdem 4 Punkte für das 1. Kriterium erhält.
Für das 3. Kriterium erhält Camunda 2 Punkte, da ein Workitem, nachdem es angenommen wurde, nicht automatisch von den Listen der anderen potenziellen User entfernt wird. Daher muss wie, in „Distribution by Offer - Multiple Resources“ detailliert beschrieben, eine Berechtigung per Code umgestellt werden, sodass die restlichen potenziellen User nicht mehr mit dem Workitem interagieren können. Demnach können sie es zwar noch sehen, aber nur der zugeteilte User kann es bearbeiten.
Daher erzielt Camunda für das Pattern „Resource-Initiated Allocation“ die Gesamtpunktzahl 8 von 10.

Für das Pattern „Resource-Initiated Execution - Allocated Work Item“, welches dem Zustandstransfer „R:start“ entspricht [93], wurden die folgenden drei Kriterien untersucht:

- 1. Die Fähigkeit einer Ressource, mit der Arbeit an einem ihr zugeteilten Workitem zu beginnen
    - Gewichtung: *1
- 2. Status des ausgewählten Workitems wird von „allocated“ auf „started“ oder äquivalent geändert
    - Gewichtung: *0,5
- 3. Es wird sichergestellt, dass das Workitem nicht von einer anderen Ressource übernommen wird
    - Gewichtung: *1

Axon Ivy erhält für alle drei Kriterien 4 Punkte, da sich diese nativ umsetzen lassen. Durch den „Start“-Button in der Weboberfläche können zugeteilte Workitems gestartet werden. Hierbei wird der Status von „OPEN (PARKED)“ auf „IN_PROGRESS(RESUMED)“ geändert. Nur der User, der das Workitem gestartet hat, sieht dieses und kann es bearbeiten. Dies ist theoretisch bereits garantiert, seitdem der User das Workitem angenommen hat, da es dadurch von der Liste der anderen potenziellen User automatisch entfernt wird, wie in „Resource-Initiated Allocation“ beschrieben.
Insgesamt erzielt Axon Ivy für das Pattern „Resource-Initiated Execution - Allocated Work Item“ die Punktzahl 10 von 10.

Für das 1. Kriterium erhält Bonitasoft 3 Punkte, da es keine echte „Start“-Funktion gibt, es aber trotzdem möglich ist, mit der Arbeit an einem zugeteilten Workitem zu einem beliebigen Zeitpunkt zu beginnen. Das Workitem wird durch die Zuteilung, wie in

„Resource-Initiated Allocation“ erklärt, direkt gestartet und die Arbeit kann zu einem beliebigen Zeitpunkt begonnen werden. Da es aber keine echte „Start“-Funktion gibt, werden nur 3 Punkte vergeben.
Bei dem 2. Kriterium erhält Bonitasoft 0 Punkte, da es keine Statusänderung gibt durch das Arbeiten an einem Workitem. Sobald das Workitem zugeteilt ist, wird der Status auf „Ready“ mit einer eingetragenen Ressource bei „Assigned to“ geändert, der in der vorliegenden Arbeit als „allocated“-Status interpretiert wird. Dieser ändert sich nicht für die weitere Bearbeitung des Workitems. Die nächste Änderung erfolgt, wenn das Workitem abgeschlossen wird. Hier wird der Status zu „Completed“ geändert. Laut [29] existiert auch ein „Executing“-Status, dieser wird aber während dem manuellen Durchlauf eines Workitems nie erreicht und kann laut [29] nur durch Code hervorgerufen werden.
Für das 3. Kriterium erhält Bonitasoft 4 Punkte, da das Workitem nur dem User angezeigt wird, der es gestartet bzw. angenommen hat. Dies ist also ab der Annahme des Workitems garantiert, da es, wie in „Resource-Initiated Allocation“ beschrieben, dann bereits von den Listen der anderen potenziellen User entfernt wird.
Bonitasoft erzielt somit für das Pattern „Resource-Initiated Execution - Allocated Work Item“ die Punktzahl 7 von 10.

Camunda erhält im 1. Kriterium, aus dem gleichen Grund wie Bonitasoft, 3 Punkte. Es gibt keine echte „Start“-Funktion und auch keinen passenden Status [52], allerdings kann die Arbeit an einem zugeteilten Workitem zu einem beliebigen Zeitpunkt gestartet werden.
Für das 2. Kriterium erhält Camunda 0 Punkte, da es keinen passenden Status gibt [52]. Es gibt nur den Status „ASSIGEND“, welchen das Workitem annimmt, nachdem es einem User zugeteilt wurde. Aus diesem Status kann das Workitem nur direkt abgeschlossen werden und in den Status „COMPLETED“ übergehen [52]. Es liegt weder eine passende „Start“-Funktion noch ein äquivalenter Zustandstransfer oder passender Status vor, wodurch signalisiert werden kann, dass ein Workitem gestartet wurde und nun daran gearbeitet wird [52].
Camunda erzielt bei dem 3. Kriterium 2 Punkte, da nur sichergestellt ist, dass das Workitem nicht von einer anderen Ressource übernommen wird, wenn die Genehmigung, welche in „Distribution by Offer - Multiple Resources“ erklärt wurde, per Code hinterlegt ist. Dadurch kann nur der User, welcher das Workitem angenommen hat, damit interagieren, wodurch das Kriterium erfüllt ist.
Zusammengefasst erzielt Camunda im Pattern „Resource-Initiated Execution - Allocated Work Item“ dadurch die Punktzahl 5 von 10.

Das Pattern „Resource-Initiated Execution - Offered Work Item“ entspricht dem Zustandstransfer „R:start_s“, wenn das Workitem nur einem User angeboten wurde und „R:start_m“, wenn es mehreren Usern angeboten wurde [93]. Für das Pattern wurden die folgenden drei Kriterien betrachtet:

- 1. Die Möglichkeit für eine Ressource, ein Workitem, das ihr angeboten wird, zu wählen und sofort mit der Arbeit daran zu beginnen
    - Gewichtung: *1
- 2. Der Status des ausgewählten Workitems wird von „offered“ auf „started“ oder äquivalent geändert
    - Gewichtung: *0,5
- 3. Es muss sichergestellt werden, dass nur eine Ressource an dem Workitem arbeiten kann (wenn es gestartet wird, darf nur noch der User, der es gestartet hat, damit interagieren)
    - Gewichtung: *1

Axon Ivy erhält für alle drei Kriterien 4 Punkte. Standardmäßig können Workitems sofort gestartet werden, ohne dass diese vorher zugeteilt bzw. angenommenen werden müssen.
Sie können direkt aus dem „OPEN (SUSPENDED)“-Status heraus gestartet werden, durch einen Klick auf den „Start“-Button in der Weboberfläche. Daraufhin wird das

Workitem in den Status „IN_PROGRESS(RESUMED)“ versetzt.
Nachdem ein User auf den „Start“-Button geklickt hat, wird das Workitem automatisch aus den Listen der anderen potenziellen User entfernt, wodurch auch das 3. Kriterium sichergestellt ist.
Axon Ivy erzielt somit die Gesamtpunktzahl 10 von 10 für das Pattern „Resource-Initiated Execution - Offered Work Item“.

Bonitasoft erhält in allen drei Kriterien 0 Punkte. Ein Workitem, welches einem User angeboten wird, muss zuerst angenommen werden, damit es bearbeitet werden kann. Dementsprechend ist es nicht möglich, ein Workitem, welches nur angeboten wird, zu starten bzw. zu bearbeiten, da es dafür erst angenommen werden muss (s. digitaler Anhang – Öffentlich). Auch Bonitasoft selbst weist den User mit folgender Fehlermeldung darauf hin: „To fill out the form, you need to take this task. This means you will be the only one able to do it. To make it available to the team again, release it.“.
Es gibt keinen „started“-Status, wie in „Resource-Initiated Execution - Allocated Work Item“ beschrieben. Zusätzlich muss das Workitem erst angenommen werden, um es auszuführen, wodurch eine direkte Statusänderung von „offered“ zu „started“ nie existieren kann.
Es ist nur möglich sicherzustellen, dass ein User an einem Workitem arbeiten kann, indem das Workitem vorher angenommen bzw. zugeteilt wurde, wie in „Resource-Initiated Execution - Allocated Work Item“ beschrieben. Da dies aber nicht möglich ist, ohne das Workitem vorher anzunehmen, kann dieser Weg nicht für das aktuelle Pattern bewertet werden. Denn in diesem Pattern soll überprüft werden, ob ein angebotenes Workitem direkt gestartet werde kann, ohne es vorher anzunehmen und dieses darf dann von keinem anderen potenziellen User mehr bearbeitet werden. Dies ist in Bonitasoft nicht möglich, ohne das Workitem vorher anzunehmen.
Dementsprechend erhält Bonitasoft für das Pattern „Resource-Initiated Execution - Offered Work Item“ den Punktestand 0 von 10. In BPMN ist das Verhalten des Pattern aufgrund der „Start“-Funktion möglich [93], welche in Bonitasoft nicht vorhanden ist, da die Workitems automatisch gestartet sind bei der Zuteilung bzw. Annahme.

Auch Camunda erzielt in allen drei Kriterien 0 Punkte. Es gibt keine Möglichkeit, ein angebotenes Workitem zu bearbeiten bzw. zu starten, da es sowohl keine „Start“-Funktion bzw. keinen passenden Status gibt als auch nur einen Zustandstransfer im Status „UNASSIGNED“ und diese ist die „claim“-Funktion, wodurch das Workitem zugeteilt werden würde [52]. Dadurch kann auch keine Statusänderung von „offered“ auf „started“ oder äquivalent ausgeführt werden. Aus dem gleichen Grund wie bei Bonitasoft ist auch hier das 3. Kriterium nicht erfüllt. In Camunda kann ein Workitem nur bearbeitet werden bzw. es kann nur gestartet werden, daran zu arbeiten, wenn das Workitem vorher angenommen bzw. zugeteilt wurde. Dementsprechend kann keines dieser Kriterien erfüllt werden.
Zusammengefasst erzielt Camunda somit für das Pattern „Resource-Initiated Execution - Offered Work Item“ den Punktestand 0 von 10. Wie bereits bei Bonitasoft erläutert, fehlt die in BPMN vorhandene „Start“-Funktion [93], um das Pattern umzusetzen.

Für das Pattern „System-Determined Work Queue Content“ wurden fünf Kriterien untersucht:

- 1.1 Die Fähigkeit des Systems, den Inhalt (Filter) zu kontrollieren, welche Workitems einer Ressource zur Ausführung vorgelegt werden
    - Gewichtung: *1
- 1.2 Die Fähigkeit des Systems, die Reihenfolge (Sortierung) zu kontrollieren, in der die Workitems einer Ressource zur Ausführung vorgelegt werden
    - Gewichtung: *1
- 2. Das System kann die Worklist nach verschiedenen Kriterien sortieren (zeitlich, nach inhaltlichen Daten, ...)
    - Gewichtung: *1

- 3.1 Das System kann die Reihenfolge individuell pro Ressource und ganzheitlich für den ganzen Prozess festlegen
    - Gewichtung: *0,5
- 3.2 Das System kann die Filter individuell pro Ressource und ganzheitlich für den ganzen Prozess festlegen
    - Gewichtung: *0,5

Für das Kriterium 1.1 erhält Axon Ivy 0 Punkte, da es nicht möglich ist, Filter vom System aus festzulegen (s. digitaler Anhang - Beantwortet). Diese können nur vom User eingestellt werden (s. digitaler Anhang - Beantwortet).
Axon Ivy erhält für das Kriterium 1.2 volle 4 Punkte, da es im Portal möglich ist, eine Sortierung vom System aus vorzugeben (s. digitaler Anhang - Beantwortet). In diesem Portal kann in den Einstellungen sowohl das Standardfeld „Portal.Tasks.SortField“, nach dem sortiert werden soll, als auch die Reihenfolge der Sortierung in dem Feld „Portal.Tasks.SortDirection“ angepasst werden (s. digitaler Anhang - Beantwortet).
Axon Ivy erhält auch für das 2. Kriterium 4 Punkte, da das Feld bzw. Kriterium festgelegt werden kann, nach dem die Workitems sortiert werden sollen, wie im Kriterium 1.2 erläutert.
Für das Kriterium 3.1 erzielt Axon Ivy 0 Punkte, da die Sortierung nur für das gesamte System und somit für alle User und alle Prozesse festgelegt werden kann (s. digitaler Anhang - Beantwortet).
Auch das Kriterium 3.2 wird in Axon Ivy mit 0 Punkten bewertet, da, wie für das Kriterium 1.1 beschrieben, Filter nicht vom System aus gesetzt werden können.
Abschließend erhält Axon Ivy für das Pattern „System-Determined Work Queue Content“ die Punktzahl 5 von 10.

Bonitasoft erzielt in allen Kriterien 0 Punkte. Es konnte keine Möglichkeit identifiziert werden, die Standardsortierung anzupassen bzw. vom System aus Filter oder Sortierungen zu hinterlegen. Auch auf Nachfrage konnte keine Antwort gefunden werden, weshalb entsprechend keine Punkte vergeben werden (s. digitaler Anhang - Unbeantwortet). Somit erhält Bonitasoft für das Pattern „System-Determined Work Queue Content“ den Gesamtpunktestand 0 von 10.

Für das 1.1 Kriterium erhält Camunda 4 Punkte, da der Administrator verschiedene Filter definieren kann und diese bestimmten Usern zur Verfügung stellen kann. Gleichzeitig kann der Zugriff auf den Standardfilter „ALL TASK“ eingeschränkt werden für diese User, sodass sie nur die entsprechenden Filter vom System her erhalten. Dies kann alles durch GUI-Elemente ohne Code erfolgen.
Bei dem Kriterium 1.2 erhält Camunda 0 Punkte, da das System die Sortierungen, im Gegensatz zu den Filtern, nicht für alle User speichern bzw. bestimmten Usern vorgeben kann (s. digitaler Anhang - Beantwortet).
Dementsprechend erhält Camunda auch für die Kriterien 2. und 3.1 jeweils 0 Punkte, da es keine Möglichkeit gibt, die Sortierung vom System aus vorzugeben (s. digitaler Anhang - Beantwortet). Sie kann nur direkt von einem User in „Tasklist“ eingestellt werden (s. digitaler Anhang - Beantwortet).
Camunda erzielt im 3.2 Kriterium 4 Punkte, da das System die Filter sowohl einzelnen Usern als auch Gruppen zuordnen kann. Zusätzlich kann in einem Filter auch definiert werden, welche Prozesse angezeigt werden sollen. Dementsprechend sind alle Möglichkeiten des Kriteriums nativ abgedeckt.
Zusammengefasst erhält Camunda daher für das Pattern „System-Determined Work Queue Content“ den Punktestand 3,75 von 10.

Bei dem nächsten Pattern „Resource-Determined Work Queue Content“ wurden die folgenden zwei Kriterien betrachtet:

- 1. Die Möglichkeit für Ressourcen, das Format und den Inhalt der in der Workqueue zur Ausführung aufgelisteten Workitems zu spezifizieren. (Filter und Sortieroptionen)
    - Gewichtung: *1

- 2. Filteroptionen (Filter und Sortieroptionen) sind entweder vorübergehend Views oder können permanent abgespeichert werden
    - Gewichtung: *1

Für das 1. Kriterium erhält Axon Ivy 3 Punkte, da ein User nativ in der Weboberfläche die Liste der ihm angezeigten Workitems, hier „Tasks“ genannt, nach verschiedenen Kriterien sortieren und filtern kann. Die Sortierung erfolgt individuell von jedem User für seine eigene Liste. Dabei kann der User bei den Sortierungskriterien zwischen den Feldern „State“, „Priority“, „Name“, „Start“ also dem Startdatum samt Uhrzeit, und „Expiry“ also dem Ablaufdatum wählen. Pro Kriterium ist eine auf- oder absteigende Reihenfolge möglich. Zusätzlich können die Workitems durch die Suchfunktion gefiltert werden. Allerdings wird nicht die volle Punktzahl vergeben, da Axon Ivy nicht nach mehreren Kriterien gleichzeitig sortieren und/oder filtern kann, wie z.B. Camunda.
Auch bei dem 2. Kriterium erzielt Axon Ivy 3 Punkte, da sowohl die Sortierungen als auch die Filter nur temporär möglich sind und nicht abgespeichert werden können, wie in anderen Tools. Nachdem ein User sich ausgeloggt hat, sind die Filter nicht mehr vorhanden und Axon Ivy fällt auf die Standardsortierung zurück. Hierbei handelt es sich um die Sortierung durch das Kriterium „Start“ von dem neusten Workitem zum Ältesten.
Zusammenfassend erhält Axon Ivy für das Pattern „Resource-Determined Work Queue Content“ die Punktzahl 7,5 von 10.

Auch Bonitasoft erhält für das 1. Kriterium 3 Punkte, da ein User nativ in der Weboberfläche die Liste der ihm angezeigten Workitems, hier „Task list“ genannt, nach verschiedenen Kriterien sortieren und filtern kann. Der User kann aus vorgegebenen Kriterien wählen, welche der Kriterien als Spalten für die Workitems angezeigt werden sollen und die Liste nach den Kriterien „Task name“, „Case ID“, „Last Update“, „Due Date“ und „Priority“ sortieren. Pro Kriterium ist auch hier eine auf- oder absteigende Reihenfolge möglich. Aufgrund des Suchmechanismus können die Workitems auch gefiltert werden. Die Sortierung und Filter legt der User individuell für die eigene Liste fest. Aus den gleichen Gründen wie bei Axon Ivy wird keine volle Punktzahl vergeben.
Bonitasoft erzielt im 2. Kriterium 3 Punkte, da die Filter und Sortierungen nicht abgespeichert werden können, wie bei Camunda. Auch hier sind nach dem Ausloggen des Users keine Filter mehr vorhanden und das Tool fällt auf die Standardsortierung zurück, also aufsteigend nach dem Kriterium „Task name“.
Bonitasoft erzielt somit für das Pattern „Resource-Determined Work Queue Content“ den Punktestand 7,5 von 10.

Camunda erzielt im 1. Kriterium 4 Punkte, da Camunda eine breite Anzahl an Filtern und Sortieroptionen zur Verfügung stellt, welche ein User nativ in der Weboberfläche nutzen kann, um die Liste der ihm angezeigten Workitems, hier „Tasklist“ genannt, nach verschiedenen Kriterien zu sortieren und zu filtern. Ein User kann Filter erstellen, wenn er die benötigte Berechtigung besitzt. In einem Filter kann ein User beliebig viele Kriterien hinterlegen, nach denen gefiltert werden soll. Auch kann angegeben werden, ob in dem Filter nur Workitems angezeigt werden sollen, die alle oder mindestens eins der Kriterien erfüllen. Auch bei den Sortierungen können beliebig viele Kriterien eingegeben werden, nach denen sortiert werden soll, wobei diese nacheinander angewendet werden. Zuerst wird nach dem Kriterium eins sortiert, innerhalb der sich ergebenden Sortierungen wird dann nach Kriterium zwei sortiert und so weiter. Je Kriterium kann auch ausgewählt werden, ob auf- oder absteigend sortiert werden soll. Sowohl bei den Filtern als auch bei der Sortierung sind die Kriterien verschiedene Felder, wie z.B. „Created“, „Due Date“, „Follow Up Date“, „Assignee“, „Priority“, „Task Name“ samt inhaltlicher Variablen aus dem Workflow.
Auch für das 2. Kriterium erhält Camunda 4 Punkte, da die Filter persistent abgespeichert werden können und auf diese Weise immer wieder verwendet werden können. Die Sortierung innerhalb der Filter kann allerdings nicht abgespeichert werden. Camunda bietet als einziges Tool die Möglichkeit, die Filter abzuspeichern.
Insgesamt erzielt Camunda für das Pattern „Resource-Determined Work Queue Content“ daher den Punktestand 10 von 10.

Für das Pattern „Selection Autonomy“ wurden die folgenden drei Bewertungskriterien untersucht:

- 1. Die Möglichkeit für Ressourcen, ein Workitem auf Grundlage dessen Eigenschaften und ihrer eigenen Präferenzen zur Ausführung auszuwählen
  - Gewichtung: *1
- 2. Es ist möglich, entweder mehrere Workitems gleichzeitig auszuführen, sodass der User jederzeit selbst bestimmen kann, welches Workitem als nächstes abgearbeitet werden soll oder nur ein Workitem gleichzeitig zu bearbeiten, aber die Ressource kann bestimmen, welches Workitem sie danach abarbeiten will
  - Gewichtung: *1
- 3. Das System kann einem User mitteilen, wenn es wichtige Workitems gibt
  - Gewichtung: *0,5

Axon Ivy erhält für das 1. und 2. Kriterium jeweils 4 Punkte, da ein User standardmäßig im Tool frei wählen kann, welches Workitem ausgeführt werden soll und mehrere Workitems gleichzeitig ausführen kann. Mehrere Workitems können allerdings nur in der Developer Workflow UI gleichzeitig ausgeführt werden. Im Portal wird ein Workitem sofort beendet und bleibt nicht im Status „IN_PORGRESS“, sobald aus dem Workitem herausgeklickt wird. Es ist allerdings bei beiden Optionen möglich, jederzeit ein Workitem abzubrechen und ein anderes mit der „Start“-Funktionalität anzufangen. Dadurch ist sichergestellt, dass der User entscheiden kann, welches Workitem er als nächstes ausführen will.
Auch in dem 3. Kriterium erzielt Axon Ivy 4 Punkte, da mithilfe der Felder „Priority“ und „Timeout“ nativ dem User mitgeteilt werden kann, welche Workitems wichtiger als die anderen sind und ggf. zuerst ausgeführt werden sollen. Die Priorität wird in der Liste der Workitems angezeigt und visualisiert sofort die Wichtigkeit eines Workitems. Zusätzlich kann im „Task“-Tab der User Task unter „Expiry“ bei „Timeout“ ein Ablaufdatum oder -zeitpunkt hinterlegt werden. Auf diese Weise ist es möglich, sowohl anzugeben, bis wann ein Workitem fertiggestellt sein sollte als auch, welcher ggf. neue Verantwortliche das Workitem übernehmen soll, falls es am Ablaufdatum nicht fertiggestellt wurde.
Abschließend erzielt Axon Ivy somit für das Pattern „Selection Autonomy“ die Gesamtpunktzahl 10 von 10.

Das 1. Kriterium erhält bei Bonitasoft 4 Punkte, da der User standardmäßig im Tool beliebig auswählen kann, welches Workitem er als nächstes abarbeiten möchte. Einem User kann die Reihenfolge nicht vorgegeben werden.
Auch für das 2. Kriterium erhält Bonitasoft 4 Punkte, da es, wie in „Resource-Initiated Allocation“, keine echte „Start“-Funktion gibt und somit ein Workitem automatisch gestartet ist, sobald es einem User zugeteilt bzw. von diesem angenommen wurde. Demnach sind alle Workitems gleichzeitig gestartet und ein User kann beliebig wählen, welches er als nächstes bearbeiten bzw. abschließen möchte. Es gibt keine Möglichkeit, eine Reihenfolge festzulegen, in der der User seine Workitems abarbeiten soll. Diese Entscheidung obliegt dem User. Der User kann jederzeit während der Bearbeitung ein Workitem schließen und ein anderes bearbeiten oder auch ein neues Workitem annehmen. Es ist somit sichergestellt, dass der User frei entscheiden kann, in welcher Reihenfolge und wann er ein Workitem abarbeitet.
Auch für das 3. Kriterium erhält Bonitasoft 4 Punkte, da das System mithilfe der Felder „Priority“ und „Due Date“ einem User anzeigen kann, welche Workitems wichtiger als die anderen sind und ggf. zuerst ausgeführt werden sollen. Beide Kriterien werden in der Liste der Workitems angezeigt. Für das Ablaufdatum wird Code benötigt, da allerdings Priorität bereits ausreicht, um das 3. Kriterium zu erfüllen, werden trotzdem 4 Punkte vergeben und das Ablaufdatum nur als Zusatzinformation bereitgestellt.
Bonitasoft erzielt damit für das Pattern „Selection Autonomy“ insgesamt die Punktzahl 10 von 10.

Camunda erzielt, aus den gleichen Gründen wie Bonitasoft, im 1. und 2. Kriterium jeweils 4 Punkte. Auch Camunda verfügt über keine echte „Start“-Funktion, wie in „Resource-

Initiated Allocation“ beschrieben und somit sind alle zugeteilten Workitems automatisch gestartet und können bearbeitet werden. Außerdem kann nativ der User frei entscheiden, welche seiner zugeteilten Workitems er in welcher Reihenfolge und wann abarbeiten möchte.
Auch für das 3. Kriterium erhält Camunda 4 Punkte, da das System mithilfe der Felder „Priority“, „Due Date“ und „Follow Up Date“ einem User anzeigen kann, welche Workitems wichtiger als die anderen sind und ggf. zuerst ausgeführt werden sollen. Alle drei Kriterien werden in der Liste der Workitems angezeigt. Die Priorität ist ein frei wählbarer Integer-Wert, der bei 0 startet. Je höher die Zahl, je wichtiger das Workitem. Falls keine Priorität angeben wird, hinterlegt Camunda den Standardwert 50.
Demnach erhält Camunda für das Pattern „Selection Autonomy“ auch die Punktzahl 10 von 10.

| | Axon Ivy | Bonitasoft | Camunda | YAWL |
|---|---|---|---|---|
| **21. Resource-Initiated Allocation** | 10 | 10 | 8 | 10 |
| **22. Resource-Initiated Execution - Allocated Work Item** | 10 | 7 | 5 | 10 |
| **23. Resource-Initiated Execution - Offered Work Item** | 10 | 0 | 0 | 10 |
| **24. System-Determined Work Queue Content** | 5 | 0 | 3,75 | 0 |
| **25. Resource-Determined Work Queue Content** | 7,5 | 7,5 | 10 | 3,75 |
| **26. Selection Autonomy** | 10 | 10 | 10 | 10 |

Abbildung 9: Ergebnisse Pull Patterns

### 3.2.4 Untersuchung der Detour Patterns

Das Pattern „Delegation“ entspricht dem Zustandstransfer „R:delegate“ [93]. Für das Pattern wurden drei Bewertungskriterien betrachtet:

- 1. Die Möglichkeit für eine Ressource, ein nicht gestartetes Workitem, das ihr zuvor zugeteilt wurde (aber noch nicht begonnen wurde), einer anderen Ressource zuzuteilen
  - Gewichtung: *1
- 2. Das Workitem wird, von der Ressource initiiert, aus der eigenen Worklist entfernt und in die Worklist der anderen Ressource ergänzt (sodass die ursprüngliche Ressource das Workitem nicht mehr auswählen bzw. damit interagieren kann)
  - Gewichtung: *0,5
- 3. Der Fall, dass ein Workitem einer Ressource zugeteilt wird, der die benötigten Berechtigungen fehlen, kann behandelt werden
  - Gewichtung: *0,5

Axon Ivy erhält für das 1. Kriterium 3 Punkte, da nativ eine „Delegate“-Funktion vorhanden ist, diese aber das Workitem einem anderen User nur anbieten und nicht zuteilen kann. Außerdem kann ein User diese Funktion bereits nutzen, bevor er das Workitem angenommen hat, während es ihm also nur angeboten wird. Über die „Delegate“-Funktion kann ein Workitem einem anderen User oder einer anderen Rolle angeboten werden, indem diese per GUI aus einer Liste ausgewählt werden.
Bei dem 2. Kriterium erzielt Axon Ivy 4 Punkte, da das Workitem durch die „Delegate“-Funktion automatisch aus der Liste des zugeteilten Users entfernt und dem neuen User oder der neuen Rolle angeboten wird. Der ursprüngliche User kann das Workitem nicht mehr sehen und es wird in der Liste der neuen potenziellen User ergänzt.

Für das 3. Kriterium erhält Axon Ivy 0 Punkte, da ein User, durch die „Delegate“-Funktion automatisch sämtliche Berechtigungen für das Workitem erhält, auch wenn dieser die Berechtigungen nicht erhalten soll. Der User wird als „Responsible“ in dem Workitem eingetragen und verfügt damit über sämtliche Rechte über das Workitem. Dasselbe gilt für eine Rolle, an die delegiert wird. Ein User, welcher nicht in der Lage sein soll, das Workitem zu bearbeiten, kann nicht aus der Liste an User bzw. Rollen entfernt werden, an die ein Workitem potenziell delegiert werden kann. Um anzupassen bzw. vorzugeben, an welche User bzw. Rollen das Workitem delegiert werden kann, muss die gewünschte Logik individuell programmiert werden, wie im Forenbeitrag für „Deallocation“ beschrieben (s. digitaler Anhang - Beantwortet). Dies wird auch in der Dokumentation für Axon Ivy benannt [9]. Dementsprechend kann der Fehlerfall, dass ein Workitem an einen User delegiert wird, der nicht die passenden Berechtigungen besitzt, nicht vorkommen, allerdings kann auch eine Delegation an einen solchen User nicht verhindert werden. Demnach kann der Fehlerfall, dass ein Workitem an einen User delegiert wird, welcher über keine passenden Berechtigungen verfügt bzw. verfügen soll nur durch individuell programmierte Logik behandelt werden.
Insgesamt erhält Axon Ivy für das Pattern „Delegation“ somit den Punktestand 6,25 von 10.

Bonitasoft erhält in allen drei Kriterien jeweils 0 Punkte, da keine Funktionalität vorliegt, um ein Workitem zu delegieren. Es gibt keine Möglichkeit für einen User, ein zugeteiltes Workitem einem anderen User anzubieten oder zuzuteilen. Auch auf Nachfrage konnte keine Antwort auf dieses Problem gefunden werden (s. digitaler Anhang - Unbeantwortet).
Bonitasoft erzielt daher für das Pattern „Delegation“ die Gesamtpunktzahl 0 von 10. Eine Funktionalität, welche in BPMN über eine „Delegate“-Funktion vorhanden ist [93], kann in Bonitasoft nicht abgebildet werden.

Camunda erhält im 1. Kriterium 3 Punkte, da der neue Username für die Delegation als String eingetragen werden muss und somit Potenzial für Rechtschreibfehler vorhanden ist, was in Axon Ivy durch das Wählen aus einer Liste besser gelöst ist. Ein User kann ein zugeteiltes Workitem delegieren, indem in der Weboberfläche ein anderer User im „Assignee“-Feld des jeweiligen Workitems eingetragen wird. Dies ist nur bei zugeteilten Workitems möglich und das Workitem wird dem neuen User auch im „ASSIGNED“-Status zugeteilt. Diese Funktionalität wird in Camunda „Reassign“ genannt. Camunda verfügt auch über eine „Delegation“-Funktion, diese kann aber nur per API angesprochen werden und müsste für die „Tasklist“-Weboberfläche individuell programmiert werden (s. digitaler Anhang - Öffentlich). Die „Reassign“-Funktionalität erfüllt die vorliegenden Kriterien und wird somit für das aktuelle Pattern in Camunda bewertet.
In dem 2. Kriterium erzielt Camunda 0 Punkte, da nur durch individuell programmierte Logik, dem ursprünglichen User die Berechtigung entzogen werden kann, mit dem delegierten Workitem zu interagieren. Wie in „Distribution by Offer - Multiple Resources“, muss die Berechtigung „TASK_WORK“ gesetzt werden, damit nur noch der zugeteilte User mit dem Workitem interagieren kann. Diesem muss in dem Unterpunkt „Authorizations“ der „Admin“-Weboberfläche die Berechtigung vergeben werden, eigene Workitems einem anderen User zuteilen zu können, durch die eben beschriebenen „Reassign“-Funktionalität. Dadurch kann der ursprüngliche User das Workitem delegieren, allerdings behält er die Berechtigung mit dem Workitem zu interagieren. Diese Berechtigung müsste manuell durch den Administrator bzw. in einer produktiven Umgebung durch hinterlegten Code entfernt werden.
Für das 3. Kriterium erhält Camunda 4 Punkte, da durch die in „Distribution by Offer - Multiple Resources“ beschriebene Berechtigung, explizit angegeben werden muss, an wen das Workitem delegiert werden kann. Hierbei kann sichergestellt werden, dass ein Workitem nur an passende User delegiert werden darf, indem das Delegieren nur zu diesen Usern freigegeben wird. Einem User müssen explizit die Rechte gegeben werden, ein bestimmtes Workitem oder auch alle Workitems eines Prozesses neu zuteilen zu dürfen. Dabei müssen dem User, der die Workitems delegieren darf, auch die Rechte

auf die User gegeben werden, denen das Workitem neu zugeordnet werden darf. Auf diese Weise lässt sich ausschließen, dass das Workitem den Usern zugeordnet wird, welche nicht die passenden Berechtigungen haben, da das Delegieren zu diesen Usern gar nicht erst freigegeben wird.
Zusammengefasst erzielt Camunda somit für das Pattern „Delegation“ die Punktzahl 6,25 von 10.

Bei dem Pattern „Escalation“ kann ein Workitem, durch das System initiiert, neu angeboten oder zugeteilt werden. Daher kann das Pattern vielen verschiedenen Zustandstransfers entsprechen, nämlich: „S:escalate_oo“, „S:escalate_om“, „S:escalate_ao“, „S:escalate_sm“, „S:escalate_am“, „S:escalate_mm“, „S:escalate_so“, „S:escalate_sa“ und „S:escalate_aa“ [93]. Für das Pattern wurden die folgenden drei Kriterien betrachtet:

- 1. Die Fähigkeit des Systems, ein Workitem einer anderen Ressource oder einer anderen Gruppe von Ressourcen zuzuordnen als die, denen das Workitem zuvor zugeordnet wurde, um zu versuchen, die Fertigstellung des Workitems zu beschleunigen
    - Gewichtung: *1
- 2. Die neue Zuteilung wird durch das System oder einen Administrator initiiert und das Workitem wird aus den Worklists der alten Ressourcen entfernt und den Worklists der neuen Ressourcen hinzugefügt
    - Gewichtung: *1
- 3. Nach der neuen Zuordnung ist das Workitem im Status „offered“ oder „allocated“
    - Gewichtung: *0,5

Axon Ivy erhält bei dem 1. Kriterium 4 Punkte, da dies nativ möglich ist, sowohl durch das System als auch durch einen Administrator. Der Administrator kann, die im Pattern „Delegation“ beschriebene „Delegate“-Funktion nutzen, um das Workitem einem neuen User oder einer neuen Rolle anzubieten. Der Administrator kann sämtliche Workitems einsehen und bearbeiten. Damit das System die neue Zuteilung übernimmt, kann in einer User Task, wie in „Selection Autonomy“ beschrieben, im „Task“-Tab unter „Expiry“ bei „Timeout“ ein Ablaufdatum oder -zeitpunkt und unter „Responsible“ ein User oder eine Rolle hinterlegt werden, an den das Workitem eskaliert wird nach Ablauf des Timeouts. Nach Ablauf wird das Workitem dann automatisch dem angegebenen User oder der angegebenen Rolle angeboten.
Auch für das 2. Kriterium erhält Axon Ivy 4 Punkte, da bei beiden Möglichkeiten für die neue Zuteilung, sowohl durch den Administrator als auch durch das System, das Workitem automatisch aus der Liste der ursprünglichen User entfernt und in die Liste der neuen User eingefügt wird.
In dem 3. Kriterium erzielt Axon Ivy 4 Punkte, da das Workitem den neuen Usern sowohl durch den Administrator als auch durch das System im Status „OPEN (SUSPENDED)“, also dem „offered“-Status, angeboten wird. Wie bei der „Delegate“-Funktion werden die Workitems auch bei der neuen Zuteilung, nach Ablauf des Timeouts, den Usern angeboten. Dies erfolgt nach dem gleichen Prinzip, welches in den „Distribution by Offer“-Patterns beschrieben wurde.
Zusammengefasst erhält Axon Ivy für das Pattern „Escalation“ den Punktestand 10 von 10.

Bonitasoft erzielt in allen drei Kriterien jeweils 0 Punkte, da keine native Funktionalität vorliegt, um die Kriterien zu erfüllen. Die einzige Möglichkeit, ein Workitem durch das System oder durch einen Administrator neu zuzuordnen bzw. zu eskalieren, ist die neue Zuordnung durch einen Administrator (s. digitaler Anhang - Beantwortet). Hierbei kann ein Administrator ein Workitem manuell per GUI einem neuen User zuordnen, allerdings nur einem User, der bereits in der potenziellen Usermenge für das Workitem vorhanden ist (s. digitaler Anhang - Beantwortet). Ein völlig neuer User, um die Fertigstellung des Workitems zu beschleunigen, wie z.B. der Vorgesetzte eines Users, ist somit nicht möglich (s. digitaler Anhang - Beantwortet). Da eine solche Zuordnung an komplett andere

User in diesem Pattern untersucht werden soll und sich somit auch die anderen Kriterien auf eine solche neue Zuordnung beziehen, erhalten alle Kriterien 0 Punkte. Die neue Zuordnung durch den Administrator im Tool hat die gleiche Funktion, wie wenn der User das Workitem wieder freigeben würde und dies soll in diesem Pattern nicht untersucht werden.
Insgesamt erzielt Bonitasoft in dem Pattern „Escalation“ somit den Punktestand 0 von 10.

Camunda erhält für das 1. Kriterium 3 Punkte, da dies nativ sowohl durch das System als auch durch einen Administrator möglich ist. Allerdings wird für die Neuzuordnung durch das System eine Zeile Code benötigt (s. digitaler Anhang - Öffentlich). Da aber die restlichen Elemente und die Möglichkeit durch den Administrator standardmäßig im Tool vorhanden sind und eine Möglichkeit ausreichend ist für das Kriterium bzw. für das Pattern können 3 Punkte vergeben werden. Für die Neuzuordnung durch das System kann ein Task Listener hinterlegt werden, in dem der Punkt „Timeout“ gewählt wird. In diesem Punkt kann sowohl ein Zeitintervall als auch ein festes Datum samt Uhrzeit angegeben werden. Nach Ablauf des Zeitintervalls kann der Task Listener ein Skript ausführen. In diesem Skript kann mit der Zeile „task.setAssignee(„Bob“);“ das Workitem einem anderen User zugeteilt werden. Falls das Workitem einer Gruppe oder einem anderen User angeboten werden soll, kann statt „setAssignee“ der Befehl „addCandidateGroup“ bzw. „addCandidateUser“ genutzt werden. Der Administrator kann sämtliche Workitems einsehen und bearbeiten und daher das „Assignee“-Feld oder das „Candidate Groups“-Feld in einem Workitem nutzen, um das Workitem einem anderen User zuzuteilen oder einer anderen Gruppe anzubieten.
Bei dem 2. Kriterium erzielt Camunda 2 Punkte, da das Workitem sowohl durch den Administrator als auch durch das System der Liste der neuen User hinzugefügt wird, aber nicht aus der Liste der ursprünglichen User entfernt wird. Wie in „Distribution by Offer - Multiple Resources“ beschrieben, muss die entsprechende Berechtigung durch eine Zeile Code gesetzt werden, damit die ursprünglichen User mit dem Workitem nicht mehr interagieren können. Aufgrund der Berechtigung werden hier nur 2 Punkte vergeben.
Camunda erzielt in dem 3. Kriterium 4 Punkte, da das Workitem nativ durch das „Assignee“-Feld einem User zugeteilt werden kann oder durch das „Candidate Groups“-Feld einer anderen Gruppe angeboten werden kann. Beides ist somit in Camunda standardmäßig möglich.
Camunda erzielt abschließend in dem Pattern „Escalation“ den Gesamtpunktestand 7 von 10.

Das Pattern „Deallocation“ korrespondiert zu den folgenden Zustandstransfers „R:deallocate_s“, „R:deallocate_m“ und „R:deallocate“ [93]. Für das Pattern wurden drei Bewertungskriterien untersucht:

- 1. Die Fähigkeit einer Ressource, ein ihr zugeteiltes (aber noch nicht begonnenes) Workitem abzugeben und einer anderen Ressource oder einer Gruppe von Ressourcen zur Zuordnung zur Verfügung zu stellen
    - Gewichtung: *1
- 2. Eine der folgenden Möglichkeiten ist umgesetzt:
  - Das Workitem kann einer einzelnen Ressource angeboten werden
  - Das Workitem kann mehreren Ressourcen angeboten werden
  - Das Workitem kann wieder in den Created-Zustand versetzt werden, sodass der Zuordnungsprozess von vorne beginnt
    - Gewichtung: *1
- 3. Es kann sichergestellt werden, dass das Workitem nicht der ursprünglichen Ressource erneut zugeteilt wird
    - Gewichtung: *0,5

Für das 1. Kriterium erzielt Axon Ivy 4 Punkte, da ein User ein Workitem nativ durch die „Reset“-Funktion oder die „Delegate“-Funktion abgeben kann. Durch die „Reset“-

Funktion in der Weboberfläche kann ein User ein Workitem wieder freigeben, nachdem er es angenommen hat. Auf diese Weise wird das Workitem wieder den restlichen Usern, denen es ursprünglich angeboten wurde, angezeigt und ein anderer User kann es annehmen. Dies ist auch möglich, wenn das Workitem nur einem User angeboten wurde, aber dann kann auch nur dieser User das Workitem sehen. Da die „Delegate“-Funktion ein Workitem einem anderen User oder einer anderen Rolle nur anbietet, kann sie auch für dieses Kriterium verwendet werden und erfüllt dieses.
Bei dem 2. Kriterium erhält Axon Ivy 4 Punkte, da alle Möglichkeiten, außer das Workitem in den Created-Zustand zu versetzen, durch die „Reset“-Funktion und „Delegate“-Funktion standardmäßig abgebildet werden. Wie für das 1. Kriterium beschrieben, kann ein Workitem, durch die Funktionen, sowohl einer einzelnen Ressource als auch mehreren Ressourcen angeboten werden.
Für das 3. Kriterium erhält Axon Ivy 0 Punkte, da dies nur durch individuell programmierte Logik möglich ist (s. digitaler Anhang - Beantwortet).
Insgesamt erzielt Axon Ivy in dem Pattern „Deallocation“ die Punktzahl 8 von 10.

Bonitasoft erhält für das 1. Kriterium 3 Punkte, da ein Workitem anderen Usern nur angeboten werden kann, wenn das Workitem durch Actor Mapping mehreren Usern angeboten wurde. Sollte ein Actor Filter verwendet worden sein, kann das Workitem zwar freigegeben werden durch den „Release“-Button in der Weboberfläche, allerdings wird das Workitem dann trotzdem nur dem ursprünglichen User angezeigt. Bei einem Actor Filter gibt es immer nur einen potenziellen User, dem das Workitem zugeordnet wird, durch das Freigeben, kann das Workitem auch nur diesem angezeigt werden. Für diesen Fall gibt es keine andere Möglichkeit, das Workitem anzubieten, wie im Pattern „Delegation“ beschrieben. In Axon Ivy kann in einem solchen Fall die „Delegate“-Funktion verwendet werden. Wird die „Release“-Funktion verwendet, wenn das Workitem durch das Actor Mapping mehreren Usern angeboten wurde, wird das Workitem diesen potenziellen User erneut angezeigt und ein beliebiger User kann es annehmen.
Bei dem 2. Kriterium erhält Bonitasoft 3 Punkte, da die ersten beiden Möglichkeiten durch die „Release“-Funktion möglich sind. Allerdings gibt es keine zusätzliche Funktionalität, wie in Axon Ivy, um das Workitem auch mehreren Usern anzubieten, falls ein Actor Filter zur Zuteilung verwendet wurde, daher wurde nicht die volle Punktzahl vergeben. Die Möglichkeit das Workitem in den Created-Zustand zu versetzen ist auch in Bonitasoft nicht möglich.
Bei dem 3. Kriterium erzielt Bonitasoft 0 Punkte, da durch die „Release“-Funktion das Workitem anschließend sämtlichen User angeboten wird, die potenzielle User sind. Dies schließt auch den User ein, der das Workitem wieder freigegeben hat. Dieser kann das Workitem erneut annehmen. Es liegt keine andere Funktionalität im Tool vor, um das Workitem anderen Usern anzubieten, wie im Pattern „Delegation“ beschrieben.
Zusammengefasst erhält Bonitasoft für das Pattern „Deallocation“ die Punktzahl 6 von 10.

Camunda erzielt im 1. Kriterium 4 Punkte, da ein Workitem sowohl durch die „Release“-Funktion als auch durch das „Candidate Groups“-Feld nativ anderen Usern angeboten werden kann. Ein User kann das ihm zugeteilte Workitem freigeben, indem er das „X“ neben seinem Namen im „Assignee“-Feld klickt. Anschließend wird das Workitem allen potenziellen User wieder angeboten. Außerdem kann ein User das Workitem auch Usern aus anderen Gruppen aktiv anbieten, indem er im „Candidate Groups“-Feld die Gruppe ändert oder weitere Gruppen hinzufügt. Wichtig anzumerken ist, dass, aufgrund der Berechtigung aus „Distribution by Offer - Multiple Resources“, der User in der „Admin“-Weboberfläche unter „Authorizations“ die Berechtigung erhalten muss, um Workitems wieder freigeben zu können und um das „Candidate Groups“-Feld bearbeiten zu dürfen. Dies kann aber ohne Code durch die entsprechenden GUIs der „Admin“-Weboberfläche erfolgen.
Für das 2. Kriterium erhält Camunda 4 Punkte, da durch die „Release“-Funktion und das „Candidate Groups“-Feld die ersten beiden Möglichkeiten umgesetzt werden können, wie im 1. Kriterium beschrieben. Auch in Camunda kann das Workitem nicht in den

Created-Zustand zurückversetzt werden, sodass der Zuordnungsmechanismus erneut ausgeführt wird.
Camunda erzielt in dem 3. Kriterium 0 Punkte, da weder die „Release“-Funktion noch das „Candidate Groups“-Feld sicherstellt, dass dem ursprünglichen User das Workitem nicht erneut zugeteilt werden darf. Der ursprüngliche User behält die Berechtigung auf das Workitem, wie im Pattern „Delegation“ beschrieben. Diese Berechtigung muss ihm nicht nur entzogen werden, sondern zusätzlich muss in der „Admin“-Weboberfläche unter „Authorizations“ hinterlegt werden, dass das Workitem dem User nicht mehr zugeteilt werden darf. Dies ist durch den Operator „DENY“ möglich. Allerdings müssten diese Berechtigungen variabel angelegt werden, was nur durch ein Skript und somit durch individuell programmierte Logik möglich ist.
Camunda erzielt daher in dem Pattern „Deallocation“ die Gesamtpunktzahl 8 von 10.

Das Pattern „Stateful Reallocation“ entspricht dem Zustandstransfer „R:reallocation_with_state“ [93]. Für das Pattern wurden zwei Kriterien betrachtet:

- 1. Die Fähigkeit einer Ressource, ein Workitem, das sie gerade ausführt, ohne Verlust von Zustandsdaten (eingegeben Daten) einer anderen Ressource zuzuteilen
    - Gewichtung: *1
- 2. Der Fall, dass ein Workitem einer Ressource zugeteilt wird, die keine Berechtigungen für die Ausführung und die Zustandsdaten (eingegeben Daten) des Workitems hat, kann behandelt werden
    - Gewichtung: *0,5

Axon Ivy erhält für beide Kriterien jeweils 0 Punkte. Das 1. Kriterium kann nur mithilfe von individuell programmierter Logik umgesetzt werden (s. digitaler Anhang - Beantwortet). Die Funktionalität ist nicht nativ in Axon Ivy vorhanden (s. digitaler Anhang - Beantwortet). Dementsprechend kann auch das 2. Kriterium nicht nativ im Tool auftreten und/oder verfügbar sein.
Axon Ivy erzielt somit für das Pattern „Stateful Reallocation“ die Gesamtpunktzahl 0 von 10. Die Funktionalität ist nicht vorhanden, obwohl BPMN das Pattern abbilden kann [93].

Auch Bonitasoft erhält für beide Kriterien jeweils 0 Punkte. Wie im Pattern „Delegation“ beschrieben, verfügt Bonitasoft über keine Funktionalität, mit der ein User ein Workitem einem anderen User zuteilen oder anbieten kann. Dementsprechend kann weder das Verhalten des 1. noch des 2. Kriteriums in Bonitasoft nativ umgesetzt werden.
Zusammengefasst erhält Bonitasoft daher für das Pattern „Stateful Reallocation“ die Punktzahl 0 von 10. Auch hier fehlt eine passende Funktionalität, obwohl BPMN das Pattern abbilden kann [93].

Camunda erzielt im 1. Kriterium 3 Punkte, da Formulare über einen „SAVE“-Button verfügen, um den Fortschritt zu speichern. Daher kann, wie im Pattern „Delegation“ beschrieben, ein Workitem einem anderen User zugeteilt werden und mithilfe des „SAVE“-Buttons können auch die Zustandsdaten mitgesendet werden. Allerdings existiert dieser „SAVE“-Button nur bei Formularen. Ist kein Formular hinterlegt, so können geänderte Variablen des Workitems nicht mitgesendet werden. Außerdem existiert, wie in „Resource-Initiated Allocation“ beschrieben, keine echte „Start“-Funktion, weswegen für dieses Pattern davon ausgegangen wird, dass ein Workitem, welches zugeteilt bzw. angenommen wurde, gleichzeitig gestartet wurde. Dementsprechend wurde nicht die volle Punktzahl vergeben.
Bei dem 2. Kriterium erzielt Camunda 4 Punkte, da explizit Berechtigungen vergeben werden müssen, an wen ein Workitem delegiert bzw. neu zugeteilt werden kann, wie im Pattern „Delegation“ beschrieben. Auf diese Weise lässt sich ausschließen, dass ein Workitem Usern zugeordnet wird, welche nicht die passenden Berechtigungen auf das Workitem oder die Zustandsdaten haben, da das Delegieren zu diesen Usern gar nicht erst freigegeben wird.
Abschließend erhält Camunda damit die Punktzahl 8,33 von 10 für das Pattern „Stateful Reallocation“.

Der passende Zustandstransfer zu dem Pattern „Stateless Reallocation“ lautet „R:reallocation_no_state“ [93]. Für dieses Pattern wurden die folgenden drei Kriterien untersucht:

- 1. Die Möglichkeit für eine Ressource, ein Workitem, das sie gerade ausführt, einer anderen Ressource neu zuzuteilen, ohne dass der Status (eingegeben Daten) erhalten bleibt
    - Gewichtung: *1
- 2. Das Workitem kann aus dem „started“-Status einer anderen Ressource im „allocated“-Status zugeteilt werden
    - Gewichtung: *0,5
- 3. Es wird sich um Nebeneffekte außerhalb des Systems gekümmert, wenn das Workitem durch das neue Zuordnen mehrmals ausgeführt wird, von verschiedenen Usern
    - Gewichtung: *0,5

Für das 1. Kriterium erhält Axon Ivy 4 Punkte, da ein Workitem nativ durch die „Delegate“-Funktionalität einem anderen User angeboten werden kann, ohne dass der Fortschritt bzw. die eingegebenen Daten mitgesendet werden. In der Ansicht des Workitems kann die „Delegate“-Funktion nicht verwendet werden, dafür muss das Workitem erst abgebrochen und so auf die Liste aller Workitems zurückgekehrt werden. Wie in dem Pattern „Selection Autonomy“ beschrieben, kann ein Workitem jederzeit während der Ausführung abgebrochen werden. Dabei werden alle eingegebenen Daten entfernt und das Workitem wird dem neuen User ohne diese angeboten.
Bei dem 2. und 3. Kriterium erzielt Axon Ivy jeweils 0 Punkte. Die „Delegate“-Funktionalität kann nur verwendet werden, wenn das Workitem abgebrochen wurde und kann ein Workitem anderen Usern nur anbieten und nicht direkt zuteilen. Außerdem gibt es keine Möglichkeit nativ im Tool Nebeneffekte abzufangen, die dadurch entstehen, dass die gleichen Schritte im Workitem, aufgrund der Neuzuteilung, mehrmals ausgeführt werden können.
Insgesamt erzielt Axon Ivy in dem Pattern „Stateless Reallocation“ den Punktestand 5 von 10.

Bonitasoft erhält für alle drei Kriterien jeweils 0 Punkte, da es grundsätzlich keine Möglichkeit gibt, ein Workitem einem anderen User neu zuzuteilen oder anzubieten, wie im Pattern „Delegation“ erläutert. Daher kann auch keines der Kriterien im Tool umgesetzt werden.
Zusammengefasst erhält Bonitasoft für das Pattern „Stateless Reallocation“ die Gesamtpunktzahl 0 von 10.

Camunda erhält für das 1. Kriterium 4 Punkte. Die Funktionalität, wie sie in „Stateful Reallocation“ beschrieben wurde, kann für dieses Kriterium auch verwendet werden. Der einzige Unterschied ist, dass es keine Möglichkeit gibt, ein Workitem ohne bereits gespeicherte Daten zu versenden. Daten, die durch den „SAVE“-Button gespeichert wurden, werden immer mitversendet. Für eine neue Zuteilung ohne die Daten müssen diese manuell entfernt werden und das Workitem muss erneut mit dem „SAVE“-Button gespeichert werden. Trotzdem kann die volle Punktzahl vergeben werden, da Variablen des Workitems immer ohne Änderungen neu zugeteilt werden. Auch Formulare, bei denen der „SAVE“-Button nicht betätigt wurde, werden grundsätzlich ohne Zustandsdaten neu zugeteilt.
Für das 2. Kriterium erhält Camunda 0 Punkte, da es im Tool, wie in „Resource-Initiated Execution - Allocated Work Item“ erläutert, keine echte „Start“-Funktion und auch keinen äquivalenten „started“-Status gibt. Allerdings ist es möglich, das Workitem während der Bearbeitung mithilfe des „Assignee“-Feldes einem anderen User zuzuteilen im „ASSIGNED“-Status.
Bei dem 3. Kriterium erzielt Camunda ebenfalls 0 Punkte, da es im Tool keine standardmäßige Möglichkeit gibt, um Nebeneffekte abzufangen, aufgrund des mehrfachen Ausführens einzelner Schritte eines Workitems. Dies muss mithilfe von individuell

programmierter Logik umgesetzt werden.
Abschließend erhält Camunda somit für das Pattern „Stateless Reallocation“ die Gesamtpunktzahl 5 von 10.

Das Pattern „Suspension-Resumption“ korrespondiert zu den beiden Zustandstransfers „R:suspend“ und „R:resume“ [93]. Für „Suspension-Resumption“ wurden vier Bewertungskriterien untersucht:

- 1. Die Möglichkeit für eine Ressource, die Ausführung eines Workitems anzuhalten und wiederaufzunehmen
    - Gewichtung: *1
- 2. Das Anhalten und das Wiederaufnehmen kann von der Ressource in der Web-Useroberfläche ausgeführt werden
    - Gewichtung: *1
- 3. Das pausierte Workitem bleibt in der Worklist des Users, allerdings wird der Status auf „suspended“ geändert
    - Gewichtung: *0,5
- 4. Das Workitem kann zu einem beliebigen Zeitpunkt in der Zukunft wiederaufgenommen werden
    - Gewichtung: *1

Axon Ivy erzielt in dem 1., 2. und 4. Kriterium jeweils 3 Punkte, da es keine echte Funktion gibt, um ein Workitem zu pausieren und anschließend fortzusetzen, aber die Funktionalität, ein Workitem abbrechen und jederzeit neu starten zu können, erfüllt den gleichen Zweck und wird daher hier bewertet. Diese Funktionalität wurde in „Selection Autonomy“ dargestellt. Da dies aber nicht die original gesuchte Funktionalität ist, wird nicht die volle Punktzahl vergeben. Der User kann in der Weboberfläche ein Workitem pausieren, indem er den „Canceln“-Button oder den Zurück-Pfeil klickt oder in einen anderen Tab der Weboberfläche klickt. Bei allen drei Methoden geht der Fortschritt im Workitem verloren. Das Workitem kann anschließend zu einem beliebigen Zeitpunkt in der Weboberfläche durch den „Start“-Button fortgesetzt werden. Dementsprechend wird das Workitem eher beendet und neu gestartet, trotzdem erfüllt dies die Kriterien.
Auch bei dem 3. Kriterium erzielt Axon Ivy 3 Punkte, da das Workitem bei allen beschriebenen Methoden in der Liste der Workitems des Users bleibt. Allerdings wird der Status nicht auf „suspended“ geändert, sondern wieder auf „OPEN (SUSPENDED)“ bzw. „OPEN (PARKED)“ Status geändert. Dem Status „OPEN (SUSPENDED)“ lässt sich entnehmen, dass das Workitem für das Tool automatisch als pausiert gilt, so lange nicht aktiv daran gearbeitet wird, auch wenn es dafür keine explizite Funktion gibt.
Insgesamt erzielt Axon Ivy für das Pattern „Suspension-Resumption“ die Punktzahl 7,5 von 10.

Auch Bonitasoft erhält in den Kriterien 1., 2. und 4. jeweils 3 Punkte, da keine echten Funktionen vorliegen, um das Workitem zu pausieren und anschließend fortzusetzen, aber die Funktionalität, ein Workitem jederzeit abbrechen und neu starten zu können, den gleichen Zweck erfüllt und daher hier bewertet wird. Wie bereits in „Resource-Initiated Allocation“ erläutert, verfügt Bonitasoft über keine „Start“-Funktion und das Workitem kann direkt bearbeitet werden, sobald es angenommen bzw. zugeteilt wurde. In der Weboberfläche kann ein Workitem ausgewählt werden und das entsprechende Formular wird direkt angezeigt und kann bearbeitet werden. Das Formular kann geschlossen werden, indem entweder in der Weboberfläche auf ein anderes Workitem geklickt wird oder das Workitem geschlossen wird durch den „X“- oder „Close“-Button. Dabei gehen die eingegeben Daten verloren, aber das Workitem kann jederzeit erneut ausgewählt und wieder bearbeitet werden, wie in „Selection Autonomy“ erläutert. Auf diese Weise kann jederzeit das Workitem abgebrochen und somit pausiert werden und durch einen Klick zu einem beliebigen Zeitpunkt wieder fortgesetzt werden.
Für das 3. Kriterium erhält Bonitasoft 3 Punkte, da das Workitem bei allen beschriebenen Methoden in der Liste der Workitems des Users bleibt. Allerdings wird der Status nicht auf „suspended“ geändert, sondern bleibt in „Ready“ mit dem Usernamen unter

„Assigned to“.
Zusammengefasst erhält Bonitasoft somit für das Pattern „Suspension-Resumption“ den Punktestand 7,5 von 10.

Camunda erhält für das 1. Kriterium 4 Punkte. Camunda verfügt, wie auch Bonitasoft, weder über eine explizite „Start“-Funktion noch über eine Funktion, um ein Workitem zu pausieren und anschließend fortzusetzen. Camunda verhält sich bezüglich des Startens, Pausierens und Fortsetzens gleich wie Bonitasoft. Der einzige Unterschied ist, dass Camunda bei Formularen die eingegeben Daten speichern kann. Dementsprechend ist das Schließen und spätere Öffnen eines Workitems nicht, wie in den beiden andern Tools, ein reines Beenden mit anschließendem Neustart. Dadurch, dass der Fortschritt gespeichert werden kann, wird ein Pausieren und Fortsetzen abgebildet, ohne über explizite Funktionen hierfür zu verfügen. Es verkörpert somit ein Fortsetzen und keinen Neustart.
Für das 2. und 4. Kriterium erhält Camunda auch 4 Punkte, da ein Workitem beliebig pausiert und fortgesetzt werden kann, durch den „Save“-Button und die in „Selection Autonomy“ beschriebene Möglichkeit, ein Workitem jederzeit abzubrechen und die Arbeit daran erneut zu starten. Ein Workitem wird geschlossen, indem in der Weboberfläche auf ein anderes Workitem geklickt wird und auch der „SAVE“-Button ist in der Weboberfläche vorhanden. Beides kann zu einem beliebigen Zeitpunkt erfolgen. Auch die Arbeit an dem Workitem kann jederzeit fortgesetzt werden, indem auf das entsprechende Workitem aus der Liste geklickt wird.
Bei dem 3. Kriterium erhält Camunda aber nur 3 Punkte, da, wie bei Bonitasoft, das Workitem in der Liste der Workitems des Users bleibt, der Status allerdings auf „ASSIGNED“ verbleibt und nicht auf „suspended“ geändert wird. Camunda verfügt über keinen passenden Status oder Zustandstransfer, um in einen solchen Status zu gelangen [52]. Dementsprechend kann hier nicht die volle Punktzahl vergeben werden.
Camunda erzielt daher den finalen Punktestand 9,64 von 10 für das Pattern „Suspension-Resumption“.

Das Zustandstransfer „R:skip“ entspricht dem Pattern „Skip“ [93]. Für dieses Pattern wurden die folgenden fünf Kriterien untersucht:

- 1. Die Möglichkeit für eine Ressource, ein ihr zugeteiltes Workitem zu überspringen und das Workitem als erledigt zu markieren
    - Gewichtung: *1
- 2. Eine Ressource kann ein Workitem abschließen, ohne dass sie gestartet hat am Workitem zu arbeiten (Das Workitem wird nur als „completed“ angezeigt, ohne dass eine Ausführung versucht/durchgeführt wurde)
    - Gewichtung: *1
- 3. Der Status wird von „allocated“ zu „completed“ geändert
    - Gewichtung: *0,5
- 4. Es kann mit Situationen, in denen ein Workitem übersprungen wird, welches Daten für folgende Workitems liefern sollte, umgegangen werden
    - Gewichtung: *0,5
- 5. Es kann festgelegt werden, welcher User die „Skip“-Funktionalität ausführen kann oder welche Workitems übersprungen werden können
    - Gewichtung: *0,5

Wie später im Pattern „Redo“ erläutert, kann eine Task hinterlegt werden, durch welche der User angeben kann, ob und wenn ja welches Workitem er wiederholen möchte. In einem Exclusive Gateway wird die Antwort überprüft und der entsprechende Pfad gewählt. Dieses Prinzip könnte theoretisch auch verwendet werden, um mit BPMN-Mitteln eine „Skip“-Funktionalität umzusetzen, indem der User angeben kann, ob er das nächste Workitem überspringen will. Bei „Skip“ ist es allerdings nicht sinnvoll, ein solches Konstrukt zu nutzen, da der User nicht weiß, welches Workitem als nächstes folgt und welche Arbeitsschritte er dafür ausführen müsste. Die Entscheidung, ob ein Workitem übersprungen werden soll, sollte erst erfolgen, wenn der Use das Workitem sieht und daran

arbeitet, aber diese Logik kann nicht mehr durch das beschriebene Konstrukt aus BPMN-Mitteln abgebildet werden. Daher wird ein solches Verhalten nicht auf das „Skip“-Pattern angewendet. Die Entscheidung muss durch BPMN-Mittel vor dem Workitem erfolgen und dies bildet zwar die Funktionalität des Überspringens ab, aber nicht in einem sinnvollen bzw. passenden Kontext.
Zusätzlich ist anzumerken, dass das „Skip“-Pattern in BPMN mithilfe der „skip“-Funktion abgebildet werden kann [93], aber in allen Tools nicht ohne individuell programmierte Logik umsetzbar ist.

Axon Ivy erzielt im 4. Kriterium 4 Punkte, da im „Output“-Tab einer User Task Standardwerte für die entsprechenden Daten hinterlegt werden können, die im Falle eines Überspringens des Workitems verwendet werden können (s. digitaler Anhang - Beantwortet). Bei allen anderen Kriterien erzielt Axon Ivy 0 Punkte, da es nur möglich ist ein Workitem zu überspringen, indem die gesamte Logik samt GUI-Elementen individuell programmiert wird (s. digitaler Anhang - Beantwortet). Dementsprechend muss auch individuell programmiert werden, welcher User die Funktionalität ausführen darf und welche Workitems übersprungen werden können (s. digitaler Anhang – Beantwortet). Außerdem wird das Workitem durch ein solches individuell programmiertes Überspringen trotzdem nur im Status „DESTROYED (DESTROYED)“ angezeigt, da es nicht wirklich ausgeführt wurde (s. digitaler Anhang – Beantwortet).
Insgesamt erzielt Axon Ivy in dem Pattern „Skip“ somit die Punktzahl 1,43 von 10.

Bonitasoft erhält in allen Kriterien 0 Punkte, da die Funktionalität mithilfe einer API individuell programmiert werden muss (s. digitaler Anhang – Beantwortet). Die gesamte Logik des Patterns muss programmiert werden und es wird nur eine API zur Verfügung gestellt (s. digitaler Anhang – Beantwortet).
Zusammengefasst erhält Bonitasoft daher den Punktestand 0 von 10 für das Pattern „Skip“.

Für das 4. Kriterium erhält Camunda 4 Punkte, da einer Variable bei der Erstellung im Modeler Standardwerte in dem dafür vorgesehenen Feld mitgegeben werden können. Sollte ein Workitem übersprungen werden, kann der entsprechende Standardwert für die Variable verwendet werden.
In allen anderen Kriterien erhält Camunda 0 Punkte, da es nur möglich ist, ein Workitem zu überspringen, indem die gesamte Logik samt GUI-Elementen individuell programmiert wird (s. digitaler Anhang - Beantwortet). Camunda stellt hierfür die „processInstanceModification“-API zur Verfügung, welche genutzt werden könnte (s. digitaler Anhang - Beantwortet). Trotzdem muss die gesamte Logik des Patterns, also welcher User die Funktionalität ausführen darf, welche Workitems übersprungen werden können und in welchen Status das Workitem versetzt wird, individuell programmiert werden (s. digitaler Anhang - Beantwortet).
Camunda erzielt im Pattern „Skip“ somit die Gesamtpunktzahl 1,43 von 10.

Das Pattern „Redo“ entspricht dem Zustandstransfer „R:redo“ [93]. Für das Pattern „Redo“ wurden vier Bewertungskriterien untersucht:

- 1.1 Die Möglichkeit für eine Ressource, ein Workitem, das zuvor in einem Case abgeschlossen wurde, zu wiederholen
    - Gewichtung: *1
- 1.2 Alle auf das wiederholte Workitem nachfolgenden Workitems (d. h. Workitems, die den nachfolgenden Tasks im Prozess entsprechen) müssen ebenfalls wiederholt werden. (Dadurch wird die Validität der folgenden/aufbauenden Workitems gewahrt)
    - Gewichtung: *1
- 2. Sämtliche Daten/Eingaben müssen überprüft werden, da diese nicht zurückgesetzt werden können. Daher dürfen sie auch nicht zerstört werden, bis der Prozess abgeschlossen ist
    - Gewichtung: *0,5

- 3. Nebeneffekte des doppelten Ausführens eines Workitems können kompensiert werden
    - Gewichtung: *0,5

Wie in „Skip“ angekündigt, kann das Pattern „Redo“ mithilfe von BPMN-Mitteln abgebildet werden. Dies steht im Widerspruch zu [93], welche besagt, dass BPMN das Pattern nicht abbilden kann. Aufgrund der Tatsache, dass das Pattern mithilfe von allgemeinen BPMN-Mitteln umgesetzt werden kann, ist die Umsetzung in allen drei Tools nahezu identisch. Sie erhalten somit alle die gleiche Gesamtpunktzahl für das Pattern. Daher wird im Folgenden einmal allgemein erläutert, wie das Pattern mit BPMN-Mitteln abgebildet werden kann, um im Anschluss nur kurz zu benennen, wo die Unterschiede in den Tools sind. Vorher ist klarzustellen, dass es, wie in [93] erläutert, außerordentlich komplex ist, dieses Pattern abzubilden, wenn mehrere User an einem Case gearbeitet haben. Demnach ist es eher umsetzbar, wenn nur ein User an einem Case gearbeitet hat [93], wie bei „Case Handling“. Daher ist es für die volle Punktzahl ausreichend, das Pattern für Cases umsetzen zu können, an denen nur ein User beteiligt war.

Allgemein kann an einem beliebigen Punkt des Prozessmodells eine zusätzliche Task eingefügt werden, durch welche die Funktionalität geboten wird, ein vorheriges abgeschlossenes Workitem zu wiederholen. Hierfür wird der Task ein Formular hinzugefügt, in welchem der User einen Wert eingeben kann. Dieser Wert korrespondiert zu einem zuvor abgeschlossenen Workitem, welches der User wiederholen möchte. Am simpelsten kann im Formular ein Text hinterlegt werden, welches Wort der User für welches Workitem eingeben muss. Es kann auch ein Dropdown oder ähnliches hinterlegt werden, dies ist in der Formularerstellung allerdings aufwendiger. Hinter der Task wird ein Exclusive Gateway eingefügt, welches die Variable, auf die der Wert aus dem Formular gespeichert wird, überprüft. In dem Exclusive Gateway ist hinterlegt, welcher Wert welchem Pfad entspricht. Aus diesem Exclusive Gateway führen dann die entsprechenden Pfade zu den korrespondierenden Tasks. Auf diese Weise können die ersten drei Kriterien umgesetzt werden.
Für das 1.1 Kriterium erhalten alle Tools 3 Punkte, da es zwar möglich ist, ein abgeschlossenes Workitem zu wiederholen, aber dafür eine extra Task eingefügt werden muss und die zu wiederholenden Möglichkeiten fest modelliert werden müssen.
Alle Tools erhalten für das 1.2 Kriterium 4 Punkte, da BPMN immer sequenziell fortschreitet und somit alle Workitems, die auf das wiederholte Workitem folgen, auch wiederholt bzw. erneut durchlaufen werden müssen.
Bei dem 2. Kriterium erzielen alle Tools 4 Punkte, da in allen Tools sämtliche eingegebenen Werte in den Workitems hinterlegt bleiben. Wie für das 1.2 Kriterium erläutert, werden alle folgenden Workitems wiederholt, wodurch alle eingegebenen Werte automatisch erneut überprüft werden.
Für das 3. Kriterium erhalten alle Tools 0 Punkte, da es keine native Möglichkeit gibt, Nebeneffekte, aufgrund des mehrfachen Ausführens der Workitems, abzufangen. Dies muss mithilfe von individuell programmierter Logik umgesetzt werden.
Insgesamt erhalten alle Tools somit die Gesamtpunktzahl 7,5 von 10 für das Pattern „Redo“.

Die Unterschiede in den Tools sind die Erstellung des Formulars und das Anlegen der Variable, welche im Pattern „Deferred Distribution“ detailliert erläutert wurden. Das Vorgehen für das Formular und die Variable ist für dieses Pattern gleich. Das GUI-Design für das Exclusive Gateway ist in allen Tools unterschiedlich, allerdings ist die Logik dahinter und wie damit interagiert werden muss überall gleich. Die Überprüfung der Variable wird entweder in einer Tabelle dargestellt oder muss einzeln in den abgehenden Pfeilen hinterlegt werden. Bei beiden Optionen muss die Überprüfung in folgender Form hinterlegt werden: „Variable == Wert“. Alles andere ist nahezu identisch.

Für das Pattern „Pre-Do“ wurden die folgenden vier Bewertungskriterien untersucht:

- 1.1 Die Möglichkeit für eine Ressource, ein Workitem vor dem Zeitpunkt auszuführen, an dem es Ressourcen angeboten oder zugeteilt wurde, die an einem bestimmten Fall arbeiten
    - Gewichtung: *1
- 1.2 Nur Workitems, die nicht von Datenelementen (und dem Ergebnis) vorangegangener Workitems abhängig sind, können vorgezogen abgearbeitet werden
    - Gewichtung: *1
- 2. Der User kann durch das vorgezogene Abarbeiten entscheiden, in welcher Reihenfolge er jegliche Workitems bearbeitet (nicht das Prozessmodell)
    - Gewichtung: *1
- 3. Alle Daten, die von mehreren Workitems verwendet werden, müssen zu Beginn des Cases existieren (demnach auch die Workitems an sich)
    - Gewichtung: *1

Für dieses Pattern wird eine ähnliche Funktionalität wie bei dem Pattern „Early Distribution“ erfordert. Bei „Early Distribution“ sollte ein Workitem angekündigt oder sogar zugeteilt werden, bevor der Prozess an der entsprechend Stelle bzw. Task angekommen ist. Bei „Pre-Do“ wird gefordert, dass das Workitem sogar ausgeführt werden kann, bevor der Prozess an der zugehörigen Task angelangt ist. Dies erfordert, dass die User das Workitem sehen können, bevor der Prozess bei der Task angekommen ist. Somit baut dieses Pattern in gewisser Weise auf der Funktionalität von „Early Distribution“ auf. Für alle drei Tools konnte das Pattern „Early Distribution“ und die geforderte Funktonalität nicht umgesetzt werden. Dementsprechend kann auch „Pre-Do“ in allen drei Tools nicht abgebildet werden, wie in den jeweiligen Absätzen in „Early Distribution“ erläutert. Bei Bonitasoft wurde auf Nachfrage keine Antwort erhalten und bei Axon Ivy und Camunda wird das Workitem erst erzeugt und dementsprechend auch hinterlegter Code erst ausgeführt, wenn der Prozess an der entsprechenden Stelle bzw. Task angekommen ist (s. digitaler Anhang – Beantwortet & Unbeantwortet).
Alle Tools erhalten somit für das Pattern „Pre-Do“ die Gesamtpunktzahl 0 von 10.

Grundsätzlich ist dieses Pattern nur schwer in einem BPMN basierten Tool vorstellbar, da es, wie im 2. Kriterium beschrieben, erfordert, dass der User durch das vorgezogene Abarbeiten die Reihenfolge der Workitems in einem Case bestimmt und nicht das Prozessmodell. Dies steht im direkten Widerspruch zur grundsätzlichen Funktionalität und Logik von BPMN, in der ein Workitem innerhalb eines Cases nur existiert, angezeigt und ausgeführt wird, wenn das Workitem der jeweils vorherigen Task und somit die Workitems aller vorherigen Tasks, abgearbeitet wurden [93, 109].

| | Axon Ivy | Bonitasoft | Camunda | YAWL |
|---|---|---|---|---|
| **27. Delegation** | 6,25 | 0 | 6,25 | 10 |
| **28. Escalation** | 10 | 0 | 7 | 8 |
| **29. Deallocation** | 8 | 6 | 8 | 10 |
| **30. Stateful Reallocation** | 0 | 0 | 8,33 | 9,17 |
| **31. Stateless Reallocation** | 5 | 0 | 5 | 6,88 |
| **32. Suspension-Resumption** | 7,5 | 7,5 | 9,64 | 10 |
| **33. Skip** | 1,43 | 0 | 1,43 | 10 |
| **34. Redo** | 7,5 | 7,5 | 7,5 | 7,5 |
| **35. Pre-Do** | 0 | 0 | 0 | 0 |

Abbildung 10: Ergebnisse Detour Patterns

### 3.2.5 Untersuchung der Auto-Start Patterns

Das Pattern „Commencement on Creation“ kann als Kombination der beiden Zustandstransfers „S:allocate“ und „S:start_on_allocate“ gesehen werden und entspricht damit

dem Zustandstransfer „S:start_on_create“ [93]. Für das Pattern wurden die folgenden zwei Kriterien betrachtet:

- 1. Die Möglichkeit für eine Ressource, mit der Ausführung eines Workitems zu beginnen, sobald das Workitem erstellt wurde
  - Gewichtung: *1
- 2. Das Workitem wird gleichzeitig erstellt, zugeteilt und gestartet
  - Gewichtung: *1

Axon Ivy erhält für beide Kriterien jeweils 3 Punkte, da beide Kriterien durch die Funktionalität „Skip Tasklist“ im „Task“-Tab einer User Task nativ abgebildet werden können, aber nur dann, wenn ein passender User bereits eingeloggt ist. Wenn das „Skip Tasklist“-Feld in einer Task angehakt ist, werden die Workitems dieser Task nach dem Erstellen direkt angeboten und gestartet. Das Formular der Task wird automatisch geöffnet und das Workitem muss nicht manuell in der Liste ausgesucht und gestartet werden. Dies geschieht aber nur, wenn der passende User bereits eingeloggt ist. Ist ein potenzieller User eingeloggt und führt den Schritt vor der Task aus, welche die „Skip Tasklist“-Funktionalität aktiviert hat, so wird das entsprechende Workitem der Task automatisch gestartet und das Formular geöffnet. Der vorherige Schritt kann sowohl das Starten des Prozesses als auch das Ausführen des vorherigen Workitems sein. Ist kein passender User eingeloggt, wird das Workitem ganz normal angeboten. Auch wenn sich anschließend ein potenzieller User einloggt, muss dieser das Workitem händisch starten. Sollte die anschließende Task auch die „Skip Tasklist“-Funktionalität aktiviert haben, wird dieses nachfolgende Workitem automatisch gestartet, da somit der Schritt vor der Task ausgeführt wurde.
Zusammenfassend erzielt Axon Ivy in dem Pattern „Commencement on Creation“ den Punktestand 7,5 von 10.

Da Bonitasoft über keine explizite „Start“-Funktion verfügt, kann das Workitem nach der Zuteilung direkt bearbeitet werden. Wie in „Resource-Initiated Allocation“ erläutert, wird dies als Starten interpretiert. Demnach wären beide Kriterien standardmäßig im Tool möglich und würden mit voller Punktzahl bewertet. Aufgrund der Tatsache, dass es aber keine echte „Start“-Funktion gibt und das Allokieren daher gleichgesetzt mit dem Starten interpretiert wird, kann nicht die volle Gesamtpunktzahl vergeben werden. Daher wird bei dem ersten Kriterium 1 Punkt abgezogen.
Bei dem 1. Kriterium erhält Bonitasoft demnach 3 Punkte. Wie bereits erläutert, wird das Workitem durch das Zuteilen automatisch auch gestartet und kann bearbeitet werden. Daher muss nur ein Actor Filter verwendet werden, sodass das Workitem bei der Erstellung automatisch einem User zugeteilt wird und dieser direkt mit der Bearbeitung starten kann, ohne es vorher annehmen oder explizit starten zu müssen. Hierfür kann ein beliebiger Actor Filter gewählt werden.
Für das 2. Kriterium erhält Bonitasoft 4 Punkte, da das Workitem durch die Verwendung eines Actor Filters gleichzeitig erstellt und zugeteilt wird und, wie beschrieben, das zugeteilte Workitem automatisch gestartet ist und bearbeitet werden kann. Dementsprechend wird das Workitem gleichzeitig erstellt, zugeteilt und gestartet.
Insgesamt erzielt Bonitasoft für das Pattern „Commencement on Creation“ damit die Gesamtpunktzahl 8,75 von 10.

Camunda verhält sich wie Bonitasoft. Auch hier gibt es, wie in „Resource-Initiated Allocation“ erklärt, keine explizite „Start“-Funktion, weswegen in Camunda ebenfalls das Zuteilen mit dem Starten gleichgesetzt wird. Somit erhält Camunda aus den gleichen Gründen wie Bonitasoft dieselben Punktzahlen: 3 Punkte für das 1. Kriterium und 4 Punkte für das 2. Kriterium. Der einzige Unterschied ist, dass in Camunda keine Actor Filter verwendet werden, um das Workitem direkt bei der Erstellung einem User zuzuteilen, sondern dass hierfür das „Assignee“-Feld verwendet wird. Dem im „Assignee“-Feld hinterlegten User wird das Workitem automatisch bei der Erstellung zugeteilt und demensprechend kann er direkt mit der Bearbeitung starten, ohne es vorher annehmen oder explizit starten zu müssen.

Zusammenfassend erhält Camunda für das Pattern „Commencement on Creation“ die Gesamtpunktzahl 8,75 von 10.

Das Pattern „Commencement on Allocation“ entspricht dem Zustandstransfer „S:start_on_allocate“ [93]. Für „Commencement on Allocation“ wurden die folgenden drei Kriterien untersucht:

- 1. Die Fähigkeit, mit der Ausführung eines Workitems zu beginnen, sobald es einer Ressource zugeteilt wird
  - Gewichtung: *1
- 2. Das Workitem wird in die Worklist des Users hinzugefügt im Status „started“ statt „allocated“
  - Gewichtung: *0,5
- 3. Das Starten wird sofort vom System getriggert, sobald das Workitem einem User zugeteilt wird (Allokation und Starten erfolgt also gleichzeitig)
  - Gewichtung: *1

Axon Ivy erzielt in allen Kriterien 0 Punkte, da eine solche Funktionalität nicht im Tool vorhanden ist (s. digitaler Anhang - Beantwortet). Die Logik und entsprechenden GUI-Elemente für die Funktionalität müssten individuell programmiert werden (s. digitaler Anhang - Beantwortet).
Insgesamt erzielt Axon Ivy in dem Pattern „Commencement on Allocation“ daher den Punktestand 0 von 10.

Bonitasoft erzielt in dem 1. und 3. Kriterium jeweils 4 Punkte, da in dem Tool, wie in „Commencement on Creation“ beschrieben, keine „Start“-Funktion vorliegt und die Workitems daher automatisch bearbeitet werden können, wenn sie zugeteilt werden. Dies wird als automatisches Starten interpretiert. Sobald ein Workitem zugeteilt bzw. angenommen wurde, kann der User sofort mit der Arbeit beginnen und muss das Workitem nicht explizit starten, da dies automatisch vom System ausgeführt wird.
Für das 2. Kriterium erhält Bonitasoft 0 Punkte, da in dem Tool kein „started“-Status vorliegt. Das Workitem bleibt im Status „Ready“ mit dem Usernamen unter „Assigned to“ während der Bearbeitung, bis es in den Status „Completed“ wechselt. Dieses Kriterium ist daher nicht möglich.
Bonitasoft erhält also für das Pattern „Commencement on Allocation“ die Gesamtpunktzahl 8 von 10. Dadurch, dass es keine „Start“-Funktion gibt, kann das Pattern im Tool, im Gegensatz zu BPMN [93], abgebildet werden.

Camunda verhält sich auch hier exakt wie Bonitasoft. Der einzige Unterschied ist, dass das Workitem in Camunda während der Bearbeitung im Status „ASSIGNED“ verbleibt, bis es in den Status „COMPLETED“ wechselt, wenn es abgeschlossen wurde. Abgesehen davon ist das Verhalten in Camunda für dieses Pattern gleich zu dem in Bonitasoft, wie auch schon in „Commencement on Creation“ beschrieben. Demnach erhält Camunda ebenfalls 4 Punkte für das 1. und 3. Kriterium und 0 Punkte für das 2. Kriterium.
Zusammengefasst erzielt Camunda den Gesamtpunktestand 8 von 10 für das Pattern „Commencement on Allocation“. Dadurch, dass es keine „Start“-Funktion gibt, kann das Pattern im Tool, im Gegensatz zu BPMN [93], umgesetzt werden.

Das Pattern „Piled Execution“ korrespondiert zu dem Zustandstransfer „R:piled_execution“ [93]. Dabei ist wichtig anzumerken, dass dieser Zustandstransfer, anders als die anderen, von einem Workitem zum nächsten springt [93]. Es verbindet die Lifecycles von zwei Workitems, in der Regel aus verschiedenen Cases [93]. Für das Pattern wurden vier Bewertungskriterien betrachtet:

- 1.1 Die Fähigkeit, das nächste Workitem einer Task (aus einem anderen Fall) zu initiieren, sobald das vorherige abgeschlossen ist, wobei alle zugehörigen Workitems derselben Ressource zugeordnet werden
  - Gewichtung: *1

- 1.2 Der Übergang zum Piled-Execution-Modus erfolgt auf Initiative einer einzelnen Ressource
    - Gewichtung: *1
- 2. Sobald ein Workitem beendet ist, wird sofort ein anderes Workitem, das derselben Task entspricht, gestartet, wenn es im Arbeitsvorrat vorhanden ist. So soll versucht werden, dass ein User immer an einem Stapel von Workitems derselben Art (also alle Workitems derselben Task) arbeitet
    - Gewichtung: *1
- 3. Es ist möglich, dass sich mehrere Ressourcen für eine bestimmte Task gleichzeitig im Piled-Execution-Modus befinden
    - Gewichtung: *0,5

Axon Ivy erhält für alle Kriterien jeweils 0 Punkte, da eine passende Funktionalität nicht vorhanden ist (s. digitaler Anhang - Beantwortet). Die gesamte Logik und GUI-Elemente müssten individuell programmiert werden (s. digitaler Anhang - Beantwortet).
Dementsprechend erzielt Axon Ivy in dem Pattern „Piled Execution“ die Punktzahl 0 von 10.

Bonitasoft erzielt ebenso in allen Kriterien jeweils 0 Punkte, da keine entsprechende Funktionalität gefunden wurde (s. digitaler Anhang - Unbeantwortet). Auch auf Nachfrage wurde keine Antwort erhalten (s. digitaler Anhang - Unbeantwortet).
Bonitasoft erhält daher für das Pattern „Piled Execution“ die Gesamtpunktzahl 0 von 10.

Camunda verfügt ebenfalls über keine passende Funktion, weswegen das Tool in allen Kriterien 0 Punkte erhält (s. digitaler Anhang - Unbeantwortet). Auch auf Nachfrage konnte keine Antwort erhalten werden (s. digitaler Anhang - Unbeantwortet).
Somit erzielt Camunda für das Pattern „Piled Execution“ den Gesamtpunktestand 0 von 10.

Das Pattern „Chained Execution“ entspricht dem Zustandstransfer „R:chained_execution“ [93]. Wie auch bei „Piled Execution“, springt „R:chained_execution“ von einem Workitem zum nächsten und verbindet die Lifecycles von zwei Workitems aus demselben Case [93]. Bei dem Pattern „Chained Execution“ wurden die folgenden fünf Kriterien untersucht:

- 1.1 Die Möglichkeit, dass das nächste Workitem in einem Case automatisch (also vom System aus) gestartet wird, sobald das vorherige Workitem abgeschlossen ist
    - Gewichtung: *1
- 1.2 Der Übergang zum Chained-Execution-Mode erfolgt auf Anweisung der Ressource
    - Gewichtung: *1
- 2. Das nächste Workitem des Cases wird nach dem Abschluss des vorherigen Workitems sofort im „started“-Status auf die Worklist der Ressource gesetzt, die den Case übernimmt
    - Gewichtung: *0,5
- 3. Die meisten (wenn nicht alle) Workitems für einen bestimmten Case müssen derselben Ressource zugeteilt und in einer strikt aufeinanderfolgenden Reihenfolge ausgeführt werden
    - Gewichtung: *1
- 4. Es kann mit dem Problem umgegangen werden, dass es zur Verzögerung von anderen Cases kommen kann, da ein Case nach dem anderen abgearbeitet werden muss
    - Gewichtung: *0,5

Axon Ivy erhält 3 Punkte für das Kriterium 1.1, da die Funktionalität durch die „Skip Tasklist“-Funktion möglich ist, wie in „Commencement on Creation“ beschrieben, außer, wenn es einen Wechsel zwischen den Usern bzw. Rollen im Prozess gibt. In einem solchen Fall muss, wie in „Commencement on Creation“ erläutert, das erste Workitem nach

dem Wechsel manuell gestartet werden und die anschließenden Workitems werden wieder automatisch gestartet.
Für das 1.2 Kriterium erhält Axon Ivy 0 Punkte, da der Haken für das „Skip Tasklist“-Feld zur Entwurfszeit in dem „Task“-Tab der User Task gesetzt werden muss. Dies kann zur Laufzeit vom User nicht geändert bzw. aktiviert werden. Dafür müssten die Logik und GUI-Elemente individuell programmiert werden.
Auch in dem 2. Kriterium erzielt Axon Ivy 3 Punkte, da dies durch die „Skip Tasklist“-Funktion möglich ist, allerdings nur so lange, bis es eine Unterbrechung gibt durch einen Userwechsel, wie für Kriterium 1.1 beschrieben. Bei einem Wechsel wird das Workitem standardmäßig im Status „OPEN (SUSPENED)“ auf die Liste des Users gesetzt. Erst nachdem der neue User dieses Workitem manuell startet und abschließt, werden die folgenden Workitems im Status „IN_PROGRESS(RESUMED)“ auf die Liste gesetzt, da sie automatisch gestartet werden.
Für das 3. Kriterium erhält Axon Ivy auch 3 Punkte, da die „Skip Tasklist“-Funktion dies ermöglicht, aber ein User, wie in „Selection Autonomy“ beschrieben, jederzeit die Möglichkeit hat, ein Workitem abzubrechen und ein anderes auszuführen. Demnach werden die Workitems bis zu einem Userwechsel in einer strikt aufeinanderfolgenden Reihenfolge ausgeführt, aber dies ist nicht verpflichtend, da der User jederzeit ein Workitem abbrechen und ein anderes aus der Liste auswählen kann.
Bei dem 4. Kriterium erzielt Axon Ivy 4 Punkte, da mit Verzögerung von anderen Cases nativ umgegangen werden kann, indem der Prozess durch die in der ersten User Task hinterlegte Rolle auf möglichste viele potenzielle User vereitelt werden kann, die einen Case des Prozesses übernehmen können. Zusätzlich kann, wie in „Selection Autonomy“ beschrieben, jederzeit ein Workitem abgebrochen werden und ein anderes bearbeitet werden, falls ein dringendes Workitem bzw. ein dringender Case in Verzug geraten würde. Außerdem kann ein User frei bestimmen, wann er einen Case übernehmen will und entsprechend Kapazität hat.
Zusammengefasst erzielt Axon Ivy in dem Pattern „Chained Execution“ die Gesamtpunktzahl 5,94 von 10.

Bonitasoft erzielt im 1.1 Kriterium 3 Punkte, da ein Workitem, wie in „Commencement on Creation“ erläutert, automatisch gestartet ist, wenn es einem User zugeteilt wurde. Die Zuteilung kann durch einen Actor Filter festgelegt werden. Hier bietet sich vor allem der Filter „Task performer“ an. Durch diesen Filter kann das aktuelle Workitem dem gleichen User zugeteilt werden, der ein vorheriges Workitem in dem Case ausgeführt hat, indem nur der Name der vorherigen Task angegeben wird. Auf diese Weise wird das nächste Workitem automatisch gestartet, sobald das vorherige Workitem abgeschlossen ist. Allerdings gibt es keine echte „Start“-Funktion, weswegen nicht die volle Punktzahl vergeben werden kann.
Auch für das 4. Kriterium erhält Bonitasoft 3 Punkte, da ein User, wie in „Selection Autonomy“ erläutert, jederzeit ein Workitem abbrechen und ein anderes bearbeiten kann, falls ein dringendes Workitem bzw. ein dringender Case in Verzug geraten ist. Außerdem kann dadurch ein User frei bestimmen, wann er entsprechend Kapazität hat und einen Case übernehmen will. Zusätzlich können auch die Workitems der ersten Task durch das Actor Mapping vielen verschiedenen potenziellen Usern zugeordnet werden, die einen Case des Prozesses übernehmen können. Dies sind dieselben Gründe wie bei Axon Ivy, allerdings konnte in Axon Ivy das gesamte Pattern umgesetzt werden, sodass die Workitems in einer strikt aufeinanderfolgenden Reihenfolge ausgeführt werden können. In Bonitasoft ist eine solche strikt aufeinanderfolgende Reihenfolge nicht möglich, weswegen nicht das gesamte Pattern abgebildet werden kann, sondern nur bestimmte Kriterien davon umgesetzt werden können. Aufgrund der Tatsache, dass das 3. Kriterium nicht umgesetzt werden kann, ist das Problem des 4. Kriterium eher irrelevant, weswegen es hier nicht mit voller Punktzahl bewertet werden kann.
Alle anderen Kriterien können in Bonitasoft nicht umgesetzt werden, weswegen das Tool jeweils 0 Punkte erzielt (s. digitaler Anhang - Unbeantwortet). Auch auf Nachfrage wurde keine Antwort erhalten (s. digitaler Anhang - Unbeantwortet). Wie in „Commencement on Allocation“ beschrieben verfügt Bonitasoft über keinen „started“-Status. Auch kann

keine strikt aufeinanderfolgende Reihenfolge vorgegeben werden, da die Workitems nur zugeteilt werden und der User, wie in „Selection Autonomy“ beschrieben, die Workitems erst auswählen muss, bevor er sie bearbeiten kann. Dies kann er jederzeit ausführen und somit Workitems in beliebiger Reihenfolge ausführen. Demnach existiert kein Chained-Execution-Mode und somit kann auch kein Übergang in diesen Modus auf Anweisung der Ressource erfolgen.
Demnach erhält Bonitasoft für das Pattern „Chained Execution“ die Punktzahl 2,81 von 10 Punkten.

Camunda verhält sich wie Bonitasoft, nur dass statt einem Actor Filter das „Assignee“-Feld genutzt wird und statt dem Actor Mapping das „Candidate Groups“- und „Candidate Users“-Feld. Ansonsten verhalten sich die Tools exakt gleich in diesem Pattern und erhalten somit auch die gleichen Punktzahlen: Für das 1.1 und 4. Kriterium jeweils 3 Punkte und für alle anderen Kriterien jeweils 0 Punkte. In Camunda müssten die Logik und GUI-Elemente für die, mit 0 Punkten bewerteten, Kriterien individuell programmiert werden (s. digitaler Anhang - Beantwortet).
Somit erzielt auch Camunda für das Pattern „Chained Execution“ den Gesamtpunktestand 2,81 von 10.

Die beiden Patterns „Chained Execution“ und „Piled Execution“ erhalten im Vergleich zu den Patterns „Commencement on Creation“ und „Commencement on Allocation“ eher geringere Punkte und Gesamtpunktzahlen. Dies ist der Fall, obwohl es bei allen vier Kriterien um das nahezu gleiche Prinzip des automatischen Startens der Workitems geht. Der Grund dafür liegt darin, dass die beiden Patterns „Commencement on Creation“ und „Commencement on Allocation“ das automatische Starten von Workitems ohne einen konkreten Use Case bzw. Grund angeben. Bei diesen Patterns wird das automatische Starten nur genutzt, damit das Workitem gestartet und bearbeitbar sein soll. Hingegen soll bei den Patterns „Chained Execution“ und „Piled Execution“ das automatische Starten verwendet werden, um sicherzustellen, dass die Workitems in einer gewissen Reihenfolge bearbeitet werden. Das System soll dadurch teilweise vorgeben können, welches Workitem als nächstes kommt. Dieses Verhalten bzw. dieser Grund für das automatische Starten wird in „Commencement on Creation“ und „Commencement on Allocation“ nie erwähnt. Daher fallen bei diesen beiden Patterns die Punktzahlen deutlich besser aus, da das hier untersuchte Starten nicht die Funktionalität beinhalten muss, den Usern vorzugeben, welches Workitem als nächstes abgearbeitet werden soll. Dies ist zusätzlich aufgrund der in „Selection Autonomy“ beschrieben Funktionalität nur schwierig in den Tools möglich, da alle User jederzeit ein Workitem abbrechen und ein beliebiges anderes Workitem auswählen können.

| | Axon Ivy | Bonitasoft | Camunda | YAWL |
|---|---|---|---|---|
| **36. Commencement on Creation** | 7,5 | 8,75 | 8,75 | 10 |
| **37. Commencement on Allocation** | 0 | 8 | 8 | 10 |
| **38. Piled Execution** | 0 | 0 | 0 | 7,14 |
| **39. Chained Execution** | 5,94 | 2,81 | 2,81 | 10 |

Abbildung 11: Ergebnisse Auto-Start Patterns

### 3.2.6 Untersuchung der Visibility Patterns

Für das Pattern „Configurable Unallocated Work Item Visibility“ wurden zwei Kriterien betrachtet:

- 1. Die Möglichkeit, die Sichtbarkeit von nicht zugeteilten Workitems für Prozessbeteiligte zu konfigurieren

    - Gewichtung: *1
- 2. Die Möglichkeit, nicht zugeteilte Workitems zu sehen, kann pro User eingestellt werden
    - Gewichtung: *0,5

Axon Ivy erhält für das 1. Kriterium 4 Punkte, da es möglich ist, die Sichtbarkeit von nicht zugeteilten Workitems sowohl durch die Rollen-Konzepte als auch durch die Genehmigungen in der „Admin"-Weboberfläche unter „Permissions" zu konfigurieren. Die Rollen können, wie in „Organisational Distribution" beschrieben, angelegt und verknüpft werden, um das gewünschte Sichtbarkeits-Szenario abzubilden. Jedem User kann anschließend individuell die gewünschte Rolle zugewiesen werden. Ein nicht zugeteiltes Workitem wird grundsätzlich allen Usern angezeigt, die in der potenziellen Usermenge vorhanden sind, da sie mindestens eine Rolle oder Unterrolle besitzen, welche für die Task angegeben ist. Zusätzlich kann unter „Permissions" für jeden User eine Berechtigung konfiguriert werden, welche die Sichtbarkeit von, für den User angebotenen oder seiner Rolle angebotenen, Workitems bestimmt. Eine solche Berechtigung ist z.B. „TaskReadAllOwnRoleWorkedOn". Dies kann nur allgemein für alle Prozesse festgelegt werden.
In dem 2. Kriterium erzielt Axon Ivy 3 Punkte, da sowohl die Rollen als auch die Genehmigungen pro User konfiguriert werden können, aber die Genehmigungen nur allgemein für sämtliche Prozesse einstellbar sind. In Camunda können die Genehmigungen pro User konfiguriert werden und sind zusätzlich variabel auf Prozesse anwendbar, statt nur allgemein für sämtliche Prozesse zu gelten. Daher erhält Axon Ivy hier nicht die volle Punktzahl.
Insgesamt erzielt Axon Ivy in dem Pattern „Configurable Unallocated Work Item Visibility" die Punktzahl 9,17 von 10.

Bonitasoft erzielt in dem 1. Kriterium 3 Punkte, da die Sichtbarkeit von nicht zugeteilten Workitems pro User nur eingestellt werden kann, indem das Actor Mapping mit den entsprechenden Usern, Rollen, Gruppen und Memberships verwendet wird. Diese können in dem Organisationsmodell, wie in „Organisational Distribution" erläutert, so angeordnet und beim Actor Mapping entsprechend zugewiesen werden, dass das gewünschte Sichtbarkeits-Szenario abgebildet ist. Allerdings verfügt Bonitasoft, wie im Pattern „Authorization" beschrieben, über keine Genehmigungen und somit auch über keine Möglichkeit, die Sichtbarkeit von nicht zugeteilten Workitems zusätzlich per Genehmigungen zu konfigurieren, wie bei den anderen Tools.
Auch für das 2. Kriterium erhält Bonitasoft 3 Punkte, da nur mithilfe des Actor Mappings die Sichtbarkeit von nicht zugeteilten Workitems pro User eingestellt werden kann, wie für das 1. Kriterium beschrieben. Es gibt keine Möglichkeit, dies auch durch Genehmigungen zu konfigurieren.
Zusammengefasst erzielt Bonitasoft in dem Pattern „Configurable Unallocated Work Item Visibility" den Punktestand 7,5 von 10.

Camunda erhält für das 1. Kriterium 4 Punkte, da die Sichtbarkeit von nicht zugeteilten Workitems sowohl durch die Felder „Candidate Groups" und „Candidate Users" als auch durch eine Vielzahl von Genehmigungen konfiguriert werden kann. In den „Candidate"-Feldern kann entweder der Usernamen oder der Gruppenname der User eingetragen werden, die das Workitem angeboten bekommen und somit im nicht zugeteilten Zustand sehen sollen. Zusätzlich kann in der „Admin"-Weboberfläche unter dem Punkt „Authorizations" hinterlegt werden, welche nicht zugeteilten Workitems ein User sehen darf. Hier stehen viele verschiedene Genehmigungen zur Verfügung, die variabel pro Prozess und pro Workitem eingestellt und sogar zur Laufzeit angepasst werden können. Auf diese Weise kann einem User oder einer Gruppe das Recht entzogen oder auch vergeben werden, konkrete Workitems zu sehen oder zusätzlich die Workitems anderer Gruppen oder User zu sehen. Das Genehmigungskonzept ist sehr ausgeprägt und flexibel einsetzbar.
Für das 2. Kriterium erhält Camunda 4 Punkte, da die Sichtbarkeit von nicht zugeteilten Workitems individuell pro User eingestellt werden kann. Einzelne User können zur

Entwurfszeit in dem „Candidate Users“-Feld hinterlegt werden. Zusätzlich können einzelne User, nachdem sie einer bestimmten Gruppe hinzugefügt wurden, sowohl zur Laufzeit als auch zur Entwurfszeit durch das „Candidate Groups“-Feld hinzugefügt werden. Auch können Genehmigungen pro User und pro Gruppe vergeben werden, um die Rechte zu vergeben oder zu entziehen, bestimmte Workitems zu sehen. Dies kann variabel für jedes einzelne Workitem eingestellt und sogar zur Laufzeit noch angepasst werden.
Camunda erzielt somit für das Pattern „Configurable Unallocated Work Item Visibility“ den Punktestand 10 von 10.

Die folgenden drei Kriterien wurden für das Pattern „Configurable Allocated Work Item Visibility“ untersucht:

- 1. Die Möglichkeit, die Sichtbarkeit von zugeteilten Workitems nach Prozessbeteiligten zu konfigurieren
    - Gewichtung: *1
- 2. Die Möglichkeit, zugeteilte (oder gestartete) Workitems zu sehen, kann pro User eingestellt werden
    - Gewichtung: *0,5
- 3. Die Möglichkeit, zugeteilte Workitems von anderen Usern zu sehen, kann konfigurieren werden (erlauben und verbieten)
    - Gewichtung: *0,5

Axon Ivy erhält für alle drei Kriterien jeweils 3 Punkte, da Sichtbarkeit von zugeteilten Workitems durch Genehmigung konfiguriert werden kann, aber dies nur sehr statisch. Nachdem ein Workitem zugeteilt wurde kann es grundsätzlich nur noch von dem User gesehen werden, der das Workitem angenommen bzw. zugeteilt bekommen hat, also dem, der im Feld „Working User“ eingetragen ist. In der „Admin“-Weboberfläche unter „Permissions“ können aber Genehmigungen wie „ReadAllTask“, „ReadAllCases“ oder „ReadTaskedTask“ vergeben werden, wodurch User mit diesen Genehmigungen die zugeteilten Workitems anderer User trotzdem sehen können. Da die Genehmigungen allerdings sehr statisch sind und daher nur allgemein festgelegt werden kann, dass ein User sämtliche zugeteilte Workitems sehen kann oder nur die eigenen, kann nicht die volle Punktzahl vergeben werden.
Diese Genehmigungen werden, wie auch bei „Configurable Unallocated Work Item Visibility“ erläutert, pro User konfiguriert.
Bei jedem User können diese Berechtigungen hinzugefügt oder auch entfernt werden. Demnach kann einem User sowohl erlaubt als auch verboten werden, die zugeteilten Workitems der anderen User zu sehen. Auch die Sichtbarkeit der eigenen zugeteilten Workitems ist mit der Berechtigung „ReadAllOwnTask“ einstellbar und kann entzogen oder vergeben werden.
Insgesamt erzielt Axon Ivy in dem Pattern „Configurable Allocated Work Item Visibility“ die Punktzahl 7,5 von 10.

Bonitasoft erziel in allen Kriterien 0 Punkte, da im Tool, wie in dem Pattern „Authorization“ beschrieben, keine Genehmigungen vorhanden sind. Demnach ist es nicht möglich, die Sichtbarkeit von zugeteilten Workitems zu konfigurieren. Nachdem ein User ein Workitem angenommen bzw. dieses zugeteilt bekommen hat, ist das Workitem nur noch für diesen User sichtbar. Da keine Genehmigungen vorliegen, um anderen Usern die Rechte zu vergeben, dieses Workitem trotzdem zu sehen, kann die Sichtbarkeit nicht eingestellt werden und das Pattern ist somit nicht abbildbar.
Daher erzielt Bonitasoft für das Pattern „Configurable Allocated Work Item Visibility“ die Punktzahl 0 von 10.

Camunda erhält für alle Kriterien jeweils 4 Punkte, da sich alle nativ durch das breite Genehmigungskonzept abbilden lassen. Grundsätzlich werden zugeteilte Workitems weiterhin den Usern angezeigt, die in den Feldern „Candidate Groups“ oder „Candidate Users“ hinterlegt sind. Ohne die in „Distribution by Offer - Multiple Resources“ beschriebene Berechtigung können diese User sogar weiterhin mit dem zugeteilten Workitem

interagieren. Daher ist es grundsätzlich in Camunda gegeben, dass ein User, sobald er in den „Candidate“-Feldern hinterlegt ist, die entsprechenden Workitems immer sieht, auch wenn diese einem anderen User zugeteilt sind. Zusätzlich liegt in der „Admin“-Weboberfläche unter „Authorizations“ die Möglichkeit vor, die „READ“-Berechtigung auf ein beliebiges Workitem einem beliebigen User zu vergeben oder zu entziehen. Unter „Authorizations“ kann in dem „Task“-Abschnitt eine „READ“-Berechtigung erstellt werden. Bei dieser Berechtigung muss die ID des gewünschten Workitems, hier Task-ID genannt, angegeben werden und die User-ID des Users, der das Workitem sehen soll. Hierbei ist es egal, ob das Workitem bereits einem User zugeteilt ist oder nicht, der angegebene User bekommt das Workitem mit dieser Berechtigung angezeigt. Diese kann einem User auch entzogen werden, indem die „READ“-Berechtigung gelöscht wird. Somit kann die Sichtbarkeit von zugeteilten Workitems sowohl vergeben als auch entzogen werden für beliebige User. Neben diesen variablen Genehmigungen können auch statische Genehmigungen vergeben werden, die einem User das Recht zugestehen oder entziehen, die eigenen zugeteilten Workitems oder die Workitems eines gesamten Prozesses etc. zu sehen oder zu bearbeiten. Demnach kann mithilfe der Genehmigungen die Sichtbarkeit von zugeteilten Workitems sowohl pro User als auch pro Gruppe festgelegt werden.
Zusammenfasst erzielt Camunda in dem Pattern „Configurable Allocated Work Item Visibility“ somit die Gesamtpunktzahl 10 von 10.

| | Axon Ivy | Bonitasoft | Camunda | YAWL |
|---|---|---|---|---|
| **40. Configurable Unallocated Work Item Visibility** | 9,17 | 7,5 | 10 | 10 |
| **41. Configurable Allocated Work Item Visibility** | 7,5 | 0 | 10 | 10 |

Abbildung 12: Ergebnisse Visibility Patterns

### 3.2.7 Untersuchung der Multiple-Resource Patterns

Für das Pattern „Simultaneous Execution“ wurden zwei Bewertungskriterien untersucht:

- 1. Die Möglichkeit für eine Ressource, mehr als ein Workitem gleichzeitig auszuführen
    - Gewichtung: *1
- 2. Die Entscheidung, welche Kombination von Workitems ausgeführt wird und in welcher Reihenfolge, wird der Ressource überlassen
    - Gewichtung: *1

Axon Ivy erhält in dem 1. Kriterium 4 Punkte, da ein User, wie in „Selection Autonomy“ erläutert, mehrere Workitems gleichzeitig ausführen kann. Dies ist allerdings nur in der Developer Workflow UI möglich. Im Portal wird ein Workitem sofort beendet und bleibt nicht im „IN_PORGRESS“-Status, sobald aus einem Workitem herausgeklickt wird. Der Fortschritt in einem Workitem wird allerdings nicht gespeichert, wenn ein anderes ausgeführt wird. Dies ist für das aktuelle Pattern aber nicht relevant.
Auch bei dem 2. Kriterium erzielt Axon Ivy 4 Punkte, da der User frei wählen darf, welche Kombination und Reihenfolge an Workitems er ausführen möchte. Dies wurde im Pattern „Selection Autonomy“ genauer erläutert.
Insgesamt erzielt Axon Ivy daher den Punktestand 10 von 10 für das Pattern „Simultaneous Execution“.

Auch Bonitasoft erhält für das 1. Kriterium 4 Punkte, da ein User mehrere Workitems gleichzeitig ausführen kann, wie in „Selection Autonomy“ beschrieben. Dadurch, dass Bonitasoft über keine explizite „Start“-Funktion verfügt, ist ein zugeteiltes Workitem automatisch gestartet, da der User es sofort bearbeiten kann. Demnach werden alle

zugeteilten Workitems gleichzeitig ausgeführt.
Für das 2. Kriterium erhält Bonitasoft auch 4 Punkte, da der User frei entscheiden kann, welche Kombination an Workitems er bearbeitet und in welcher Reihenfolge. In der Liste seiner Workitems kann der User dies selbst festlegen. Dies wurde in „Selection Autonomy“ erklärt. Nur der Fortschritt in einem Workitem wird nicht gespeichert, wenn er anfängt, ein anderes zu bearbeiten. Dies ist in dem aktuellen Pattern allerdings nicht relevant.
Zusammengefasst erzielt Bonitasoft daher in dem Pattern „Simultaneous Execution“ den Punktestand 10 von 10.

Camunda verhält sich für das Pattern nahezu exakt wie Bonitasoft. Der einzige Unterschied ist, dass der Fortschritt in einem Workitem mithilfe des „SAVE“-Buttons gesichert werden kann. Ansonsten verhält sich Camunda bei beiden Kriterien identisch zu Bonitasoft, wie auch in „Selection Autonomy“ beschrieben, und erhält somit auch in beiden Kriterien jeweils 4 Punkte.
Daher erzielt Camunda für das Pattern „Simultaneous Execution“ den Gesamtpunktestand 10 von 10.

Bei dem Pattern „Additional Resources“ wurden die folgenden beiden Bewertungskriterien betrachtet:

- 1. Die Möglichkeit für eine gegebene Ressource, zusätzliche Ressourcen anzufordern, die sie bei der Ausführung eines laufenden Workitems unterstützen
    - Gewichtung: *1
- 2. Mehrere Ressourcen können gleichzeitig am selben Workitem arbeiten
    - Gewichtung: *1

Axon Ivy erhält für beide Kriterien jeweils 3 Punkte, da das Portal die Möglichkeit bietet, Unteraufgaben für ein Workitem und Chats für eine Case zu erstellen (s. digitaler Anhang - Beantwortet). Diese Unteraufgaben können anderen Usern oder Rollen zugeordnet werden, um Aufgaben auszulagern, bei denen Unterstützung benötigt wird, um so zusammen an einem Workitem zu arbeiten (s. digitaler Anhang - Beantwortet). In einem Chat können beliebige User und Rollen hinzugefügt werden, um sich über ein Workitem auszutauschen und Fragen zu klären (s. digitaler Anhang - Beantwortet). Beide Funktionalitäten sind allerdings nur in dem Portal vorhanden und müssen erst aktiviert werden (s. digitaler Anhang - Beantwortet). Daher wird für beide Kriterien nicht die volle Punktzahl vergeben.
In dem Pattern „Additional Resources“ erzielt Axon Ivy somit den Punktestand 7,5 von 10.

Bonitasoft erzielt in beiden Kriterien 0 Punkte, da es nicht möglich ist, Unterstützung anzufordern oder zusammen an einem Workitem zu arbeiten (s. digitaler Anhang - Unbeantwortet). Es konnte weder eine Methode im Tool noch in der Weboberfläche identifiziert werden, die eine solche Funktionalität zur Verfügung stellt. Auch auf Nachfrage wurde keine Antwort erhalten (s. digitaler Anhang - Unbeantwortet).
Zusammengefasst erhält Bonitasoft daher für das Pattern „Additional Resources“ die Gesamtpunktzahl 0 von 10.

Auch Camunda erzielt in beiden Kriterien 0 Punkte, da ein Workitem immer nur einem User zugeteilt werden kann und daher auch immer nur ein User an einem Workitem arbeiten kann (s. digitaler Anhang - Beantwortet). Sollte ein User Unterstützung benötigen, muss der User das Workitem mit der „Reassign“-Funktion einem anderen User neu zuteilen (s. digitaler Anhang - Beantwortet). Demnach sind beide Kriterien nicht erfüllt, da Unterstützung nur durch die Neuzuteilung eines Workitems möglich ist (s. digitaler Anhang - Beantwortet) und dies nicht die geforderten Kriterien erfüllt. Eine entsprechende Funktionalität müsste dementsprechend individuell programmiert werden (s. digitaler Anhang - Beantwortet).
Insgesamt erzielt Camunda daher in dem Pattern „Additional Resources“ die Punktzahl 0 von 10.

| | Axon Ivy | Bonitasoft | Camunda | YAWL |
|---|---|---|---|---|
| **42. Simultaneous Execution** | 10 | 10 | 10 | 10 |
| **43. Additional Resources** | 7,5 | 0 | 0 | 0 |

Abbildung 13: Ergebnisse Multiple-Resource Patterns

## 3.3 Ergebnisse der Untersuchung

### 3.3.1 Vergleich der Tools aufgrund der Untersuchungsergebnisse der Patterns

Abschließend sind die Endergebnisse der Tools in der folgenden Abbildung tabellarisch zusammengefasst:

| | Axon Ivy | Bonitasoft | Camunda | YAWL |
|---|---|---|---|---|
| **1. Direct Distribution** | 5 | 10 | 7,5 | 10 |
| **2. Role-Based Distribution** | 9,17 | 10 | 7,22 | 10 |
| **3. Deferred Distribution** | 8 | 4 | 8 | 8 |
| **4. Authorization** | 5 | 0 | 6,67 | 5 |
| **5. Separation of Duties** | 0 | 0 | 0 | 8 |
| **6. Case Handling** | 8,75 | 10 | 7,5 | 10 |
| **7. Retain Familiar** | 5 | 10 | 5 | 10 |
| **8. Capability-Based Distribution** | 4,29 | 4,29 | 0 | 7,86 |
| **9. History-Based Distribution** | 0 | 0 | 0 | 3,33 |
| **10. Organisational Distribution** | 7,5 | 9,17 | 0 | 10 |
| **11. Automatic Execution** | 10 | 7,5 | 7,5 | 7,5 |
| **12. Distribution by Offer - Single Resource** | 10 | 10 | 10 | 10 |
| **13. Distribution by Offer - Multiple Resources** | 10 | 10 | 8 | 10 |
| **14. Distribution by Allocation - Single Resource** | 2,5 | 5 | 10 | 10 |
| **15. Random Allocation** | 0 | 0 | 0 | 10 |
| **16. Round Robin Allocation** | 0 | 0 | 0 | 9,17 |
| **17. Shortest Queue** | 0 | 0 | 0 | 10 |
| **18. Early Distribution** | 0 | 0 | 0 | 0 |
| **19. Distribution on Enablement** | 8,13 | 8,75 | 10 | 10 |
| **20. Late Distribution** | 8,75 | 8,75 | 8,75 | 8,75 |
| **21. Resource-Initiated Allocation** | 10 | 10 | 8 | 10 |
| **22. Resource-Initiated Execution - Allocated Work Item** | 10 | 7 | 5 | 10 |
| **23. Resource-Initiated Execution - Offered Work Item** | 10 | 0 | 0 | 10 |
| **24. System-Determined Work Queue Content** | 5 | 0 | 3,75 | 0 |
| **25. Resource-Determined Work Queue Content** | 7,5 | 7,5 | 10 | 3,75 |
| **26. Selection Autonomy** | 10 | 10 | 10 | 10 |

| | | | | |
|---|---|---|---|---|
| **27. Delegation** | 6,25 | 0 | 6,25 | 10 |
| **28. Escalation** | 10 | 0 | 7 | 8 |
| **29. Deallocation** | 8 | 6 | 8 | 10 |
| **30. Stateful Reallocation** | 0 | 0 | 8,33 | 9,17 |
| **31. Stateless Reallocation** | 5 | 0 | 5 | 6,88 |
| **32. Suspension-Resumption** | 7,5 | 7,5 | 9,64 | 10 |
| **33. Skip** | 1,43 | 0 | 1,43 | 10 |
| **34. Redo** | 7,5 | 7,5 | 7,5 | 7,5 |
| **35. Pre-Do** | 0 | 0 | 0 | 0 |
| **36. Commencement on Creation** | 7,5 | 8,75 | 8,75 | 10 |
| **37. Commencement on Allocation** | 0 | 8 | 8 | 10 |
| **38. Piled Execution** | 0 | 0 | 0 | 7,14 |
| **39. Chained Execution** | 5,94 | 2,81 | 2,81 | 10 |
| **40. Configurable Unallocated Work Item Visibility** | 9,17 | 7,5 | 10 | 10 |
| **41. Configurable Allocated Work Item Visibility** | 7,5 | 0 | 10 | 10 |
| **42. Simultaneous Execution** | 10 | 10 | 10 | 10 |
| **43. Additional Resources** | 7,5 | 0 | 0 | 0 |
| **Finale Bewertung:** | 247,88/43 = ca. 5,76 | 200,02/43 = ca. 4,65 | 235,6 /43 = ca. 5,48 | 350,05/43 = ca. 8,14 |

Abbildung 14: Endergebnisse

Es lässt sich der Abbildung 14 entnehmen, dass Axon Ivy von allen Tools die Resource Patterns am besten abbilden kann, mit einer finalen Bewertung von ca. 5,76 von 10. Camunda liegt mit einer durchschnittlichen Punktzahl von ca. 5,48 knapp hinter Axon Ivy. Mit einem deutlichen Abstand konnten die Resource Patterns in Bonitasoft mit einer durchschnittlichen Punktzahl von ca. 4,65 am schlechtesten abgebildet werden.

Die Stärken von Axon Ivy liegen vor allem in zwei Funktionen: der expliziten „Start"-Funktion und dem Portal. Axon Ivy verfügt als einziges Tool über eine explizite „Start"-Funktion, mit welcher Workitems, in egal welchem Status, manuell gestartet werden können. Darüber hinaus können über den sog. Market zusätzliche Module, wie z.B. das Portal, kostenlos heruntergeladen werden. Auf diese Weise können viele verschiedene Funktionalitäten, wie z.B. Chats oder Unteraufgaben, verwendet werden. Diese Funktionen tragen maßgeblich zu den besten Bewertungen von Axon Ivy bei, was vor allem bei den Pull Patterns und Multiple-Resource Patterns sichtbar ist. Die größte Stärke von Axon Ivy liegt allerdings in der Tatsache, dass das Tool über eine sehr breite Auswahl an Funktionalitäten verfügt, auch wenn die einzelnen Funktionen nicht so tiefgreifend bzw. umfangreich sind, wie bei den anderen Tools. Die Funktionen, die bei den anderen Tools herausragen, wie bei Bonitasoft das Organisationsmodell und der Zuordnungsmechanismus und bei Camunda das Berechtigungsmanagement, sind auch in Axon Ivy möglich, nur nicht so umfangreich bzw. variabel. Dies lässt sich auch in den folgenden Funktionen erkennen: Die „Skip Tasklist"-Funktion ermöglicht zwar das automatische Starten eines Workitems bei der Erstellung, allerdings nicht das automatische Starten eines Workitems bei der Zuteilung bzw. Annahme. Dies hat z.B. die schlechte Punktzahl im Pattern „Commencement on Allocation" zur Folge. Auch sind die Standardrollen „SELF" und „CREATOR" vorhanden, um Patterns wie „Retain Familiar" oder „Case Handling" umzusetzen, allerdings sind diese nicht tiefgehend genug, um eine bessere Punktzahl als Bonitasoft zu erhalten. Mit den Standardrollen kann nur der Initiator oder der User des vorherigen Workitems gefunden werden, in Bonitasoft kann der User eines beliebigen vorherigen Workitems gefunden werden. Das letzte Beispiel ist die „Delegate"-Funktion. Diese ist in Axon Ivy nativ am besten umgesetzt, indem eine GUI mit

Dropdown-Userliste zur Verfügung gestellt wird. Allerdings kann ein Workitem standardmäßig nur angeboten und nicht zugeteilt werden und die Userliste kann nicht eingeschränkt werden, was beides in Camunda möglich ist. Die Funktionen erfüllen die Patterns teils, sind aber nicht umfangreich bzw. flexibel genug, um die beste Punktzahl zu erhalten. Axon Ivy positioniert sich somit, obwohl das Tool in den Resource Patterns die höchste Gesamtpunktzahl hat, eher als Allrounder. Dies kann auch an den Bewertungen aus der Abbildung 14 erkannt werden. Axon Ivy erzielt nur siebenmal die alleinige beste Punktzahl von den drei untersuchten Tools. Camunda erhält im Vergleich achtmal die alleinige beste Punktzahl und Bonitasoft sogar fünfmal. Alle Tools liegen hierbei nah beieinander und Axon Ivy, als insgesamt höchstbewertetes Tool, ist weniger häufig alleiniger Sieger in einem Pattern als der Zweitplatzierte, Camunda. Dass Axon Ivy am besten bewertet wurde, liegt somit nicht an bestimmten überragenden Funktionen, sondern an einer breiten Aufstellung. Falls Axon Ivy in einem Pattern nicht am besten bewertet wurde, so ist die Punktzahl von Axon Ivy selten weit von der besten Bewertung entfernt. Die einzigen Ausnahmen sind die Patterns, in denen Axon Ivy 0 Punkte erhält. Dies erfolgt sehr selten und mit Ausnahme der Patterns, in denen alle Tools 0 Punkte erzielen, ist Axon Ivy nur einmal alleiniges Tool ohne Punkte und nur einmal zusammen mit einem anderen Tool ohne Punkte. Darüber hinaus ist Axon Ivy insgesamt nur fünfmal alleiniger Letztplatzierter bei einem Pattern. Alle anderen Tools belegen den alleinigen letzten Platz häufiger. Zusätzlich sind die Gesamtpunktzahlen von Axon Ivy bei diesen Patterns nur dreimal unter 7 Punkten.
Axon Ivy verfügt über keine besonderen Schwächen, was das Bild des Allrounders weiter festigt. Als Schwäche fällt allein auf, dass die Zuordnung der Workitems etwas starr ist, da keine Workitems nativ zugeteilt werden können und nicht mehrere Ressourcen einzeln angegeben werden können.

Bonitasoft, welches in den Resource Patterns die niedrigste Gesamtpunktzahl erzielt hat, hat seine Stärken klar in dem Organisationsmodell und den Zuteilungsmechanismen. Die im Organisationsmodell angelegten Elemente können im Actor Mapping sehr flexibel verwendet werden. Zusätzlich liegen vorgefertigte Actor Filter vor, um gewisse Szenarien nativ abbilden zu können. Dies lässt sich auch an den Punktzahlen ablesen. Bonitasoft ist häufig genau in den Patterns führend, in denen diese Funktionen verwendet wurden. Vor allem bei den Creation Patterns lassen sich die Stärken von Bonitasoft erkennen, da alle fünf Patterns, in denen Bonitasoft die alleinige beste Punktzahl erhält, zu den Creation Patterns gehören. Außerdem ist Bonitasoft bei den Auto-Start Patterns identisch zu Camunda, aufgrund der gleichen Funktionsweise ohne explizite „Start“-Funktion.
Die Schwächen von Bonitasoft liegen in fehlenden Funktionalitäten. Eine Funktion, um Workitems zu delegieren bzw. neu zuzuteilen, ist nicht vorhanden, genauso fehlt ein Berechtigungsmanagement. Ein Berechtigungsmanagement ist in der kostenpflichtigen Version vorhanden. Diese Funktionalität ist aber nicht in der Community Version beinhaltet. In den Patterns, aus der Abbildung 14, in denen Bonitasoft am weitesten hinter den anderen Tools liegt, wären diese fehlenden Funktionalitäten benötigt worden. Demnach sind die fehlenden Funktionalitäten für diese größten Punkteunterschiede verantwortlich. Zusätzlich lässt sich die schlechte Gesamtbewertung von Bonitasoft daran erkennen, dass Bonitasoft zehnmal die alleinige schlechteste Bewertung aus den drei Tools hat. Dabei sind die Gesamtpunktzahlen in diesen Patterns nur einmal höher als 7.

Camunda hat seine Stärken, als knapp Zweitplatzierter, vor allem in dem umfangreichen und flexiblen Berechtigungsmanagement und der Möglichkeit, den Fortschritt in einem Workitem abzuspeichern. Diese Funktionen wurden in den Patterns verwendet, in denen Camunda die meisten Punkte bzw. als einziges der drei Tools die höchste Punktzahl erhalten hat. Vor allem durch das Berechtigungsmanagement und den daraus resultierenden Möglichkeiten konnte Camunda konkrete Probleme der Patterns als einziges Tool nativ lösen, wie z.B. bei „Delegation“ oder „Stateful Reallocation“. Die Genehmigungen müssen häufig erstmals vergeben bzw. freigegeben werden, wodurch hier die Probleme nicht auftreten können. Die Stärken sind auch in der Abbildung 14 sichtbar. Die

hohen Punktzahlen in den Visibility Patterns konnten vor allem durch das Berechtigungsmanagement erzielt werden. Die Punkte in den Detour Patterns stammen aus der Kombination von Berechtigungen und der Fähigkeit, den Fortschritt eines Workitems abzuspeichern. Grundsätzlich liegt die Gesamtpunktzahl von Camunda nur sehr knapp hinter der von Axon Ivy. Als Zweitplatzierte erzielt Camunda sogar achtmal die alleinige beste Punktzahl, einmal mehr als Axon Ivy. Grundsätzlich hat Camunda deutlich weniger schlechte Abweichungen als Bonitasoft, indem es nur siebenmal die alleinige schlechteste Bewertung erhält und viermal davon trotzdem 7 oder mehr Punkte hat. Bonitasoft im Vergleich ist zehnmal alleiniges Schlusslicht und nur einmal davon mit 7 oder mehr Punkten. Bei den Auto-Start Patterns ist Camunda aufgrund der gleichen Funktionsweise ohne explizite „Start"-Funktion identisch zu Bonitasoft.
Allerdings hat Camunda bei dem Organisationsmodell seine Schwächen. Es liegt keine native Möglichkeit vor, um Hierarchien zu erstellen. Die User können zwar Gruppen zugeordnet werden, allerdings gibt es weder die Möglichkeit, User miteinander noch Gruppen miteinander in Beziehung zu setzen. Auch können Usern keine Fähigkeiten bzw. zusätzlichen Attribute mitgegeben werden. Dies alles spiegelt die sehr minimalistischen Möglichkeiten im Organisationsmodell wider. Zusätzlich wird ein Workitem den restlichen potenziellen Usern weiterhin angezeigt, nachdem es angenommen bzw. zugeteilt wurde. Nur durch die in „Distribution by Offer - Multiple Resources" beschriebene Berechtigung, welche per Code eingetragen werden muss, können die restlichen User nicht mehr mit dem Workitem interagieren. Vor allem wenn ein Workitem von einem User neu zugeteilt wird, behält dieser die Bearbeitungsrechte auf dem Workitem. Diese müssen per Code oder manuell entfernt werden. Diese Schwächen sind hauptverantwortlich für die Patterns, in denen Camunda die schlechtesten Punktzahlen erhalten hat, mit den größten Punktdifferenzen zu den anderen Tools.

Grundsätzlich ist anzumerken, dass bei komplexeren Patterns mit aufwendigeren Zuteilungsmechanismen, wie „Separation of Duties", „History-Based Distribution", „Random Allocation", „Round Robin Allocation", „Shortest Queue" und „Piled Execution", alle drei Tools Probleme hatten und 0 Punkte erzielt haben. Diese sind standardmäßig, ohne die jeweilige Logik zu programmieren, nicht umsetzbar in den Tools. Auch konnten die zwei Pattern „Early Distribution" und „Pre-Do", welche nicht mit der logischen Ablauffolge der Tools übereinstimmen, in keinem Tool umgesetzt werden. In fünf Patterns, welche mit der Standardfunktionalität der Tools und/oder von BPMN übereinstimmen, erhielten alle Tools dieselben Gesamtpunktzahlen. Dementsprechend wurden dreizehnmal alle Tools gleich bewertet: achtmal gleich schlecht mit 0 Punkten und fünfmal gleich gut.

YAWL hat, wie zu erwarten, die beste Gesamtbewertung aller Tools erhalten mit einer Punktzahl von ca. 8,14 und liegt somit deutlich vor den anderen betrachteten Tools. Nahezu alle Patterns waren nativ umsetzbar und erhalten hohe Punktzahlen, darunter auch 25 Patterns, die mit voller Punktzahl bewertet wurden. Dementsprechend wurden über 50% der untersuchten Patterns fehlerfrei in YAWL umgesetzt. Vor allem in den bereits angesprochenen komplexeren Patterns mit aufwendigeren Zuteilungsmechanismen, wie „Separation of Duties", „History-Based Distribution", „Random Allocation", „Round Robin Allocation", „Shortest Queue" und „Piled Execution", überzeugt YAWL mit hohen Punktzahlen, da diese nativ abgebildet werden können. Hier lässt sich der größte Unterschied zu den restlichen Tools feststellen.
Vor dem Hintergrund, dass YAWL entwickelt wurde, mit dem Ziel, in einem Tool die Workflow Patterns abbilden zu können [93, 105, 107], sind diese Ergebnisse wenig überraschend. Daher sind vor allem die Patterns interessant, in denen auch YAWL Probleme hatte. Grundsätzlich ist YAWL nur selten nicht am besten oder mit am besten bewertet, insgesamt nur achtmal. Davon erhält YAWL viermal eine Gesamtbewertung von 0 und konnte Patterns somit nicht abbilden. Darunter sind die beiden Patterns „Early Distribution" und „Pre-Do", welche kein Tool umsetzen konnte, aufgrund des beschriebenen Wiederspruchs zur logischen Ablauffolge der Tools. Ein weiteres Pattern welches YAWL gar nicht abbilden konnte, war „Additional Resources". Dieses Pattern ist auch untypisch für klassische WfMS, welche standardmäßig einem Workitem nur einen User zuteilen.

Das letzte Pattern mit einer Gesamtbewertung von 0 ist „System-Determined Work Queue Content“. Dieses ist mit das einzige Pattern, in dem YAWL, im Gegensatz zu den BPMN basierten Tools die das Pattern umsetzen konnten, keine passende Funktionalität aufweist. Diese Funktionalität „Work Queue Content“ anzupassen, sowohl durch das System als auch durch den User ist in YAWL nicht sehr ausgeprägt. Dies unterstützt auch der Fakt, dass das Pattern „Resource-Determined Work Queue Content“ das einzige ist, in dem alle anderen Tools besser bewertet sind als YAWL. YAWL verfügt hier über keine Möglichkeit die Worklist zu filtern bzw. nach Workitems zu suchen, was in allen anderen Tools standardmäßig möglich ist. Die Gesamtpunktzahl von YAWL liegt hier 6,25 Punkte hinter der Punktzahl des bestbewerteten Tools. Das einzig andere Pattern, in dem YAWL, im Gegensatz zu den BPMN basierten Tools die das Pattern umsetzen konnten, eine Funktionalität nicht aufweist ist „Escalation“. In YAWL kann nur ein Administrator ein Workitem einer komplett neuen Ressource zuordnen, die vorher nicht als potenzielle Ressource der Task eingetragen war. Eine Möglichkeit, dass dies nicht manuell, sondern durch das System ausgeführt werden soll, liegt nicht vor. Trotzdem liegt YAWL hier nur 2 Punkte hinter dem bestbewerteten Tool. Bei den restlichen zwei Patterns, in denen YAWL nicht am besten oder mit am besten bewertet wurde, lag der Grund darin, dass die Funktion in dem bestbewerteten Tool umfangreicher war. Bei dem Pattern „Authorization“ verfügt Camunda über flexiblere und eine größere Auswahl an Berechtigungen als YAWL und ist daher besser bewertet. In dem Pattern „Automatic Execution“ war Axon Ivy besser, da es über vorgefertigte Task-Typen verfügt, um gewisse automatische Funktionen auszuführen, wie Verbindungen zu Datenbanken oder Web-Services oder dem Absenden einer E-Mail. Bei „Authorization“ liegt YAWL nur 1,67 Punkte und bei „Automatic Execution“ nur 2,5 Punkte hinter den jeweils bestbewerteten Tools. Selbst in den vier Patterns in denen YAWL das Pattern zwar umsetzen konnte, also keine 0 Punkte erhalten hat aber nicht am besten bewertet wurde, ist YAWL häufig nicht weit von der besten Gesamtbewertung in dem Pattern entfernt. Abgesehen von den aufgezählten Patterns, hat YAWL in allen restlichen Patterns die höchste oder mit die höchste Gesamtpunktzahl. Punktverluste in diesen Patterns erfolgen häufig nur, da für einzelne Kriterien Fehlerfälle behandelt bzw. abgefangen werden sollen und dies häufig nur durch Code möglich ist, aufgrund der vielen verschiedenen Fälle.
Die detaillierte Übersicht über die Punktzahlen und wie die einzelnen Patterns bzw. Kriterien in YAWL umgesetzt wurden, sind, wie bereits erläutert, in der Datei „Detaillierte_Bewertungsübersicht_der_Patterns.xlsx“ im digitalen Anhang zu finden.

### 3.3.2 Bewertung der Antworten auf Forenbeiträge

Die Antworten auf Forenbeiträge waren sehr hilfreich und meist qualitativ hochwertig, allerdings wurden eher selten neue unbekannte Aspekte dargestellt. Die Antworten auf die Forenbeiträge haben die vorher vermutete Kategorie und dementsprechend auch die Punktzahl nur belegt und nie verändert, bis auf zwei Ausnahmen: Die Patterns „Additional Resources“ und „System-Determined Work Queue Content“ in Axon Ivy, bei welchen die Antworten der Forenbeiträge auf Funktionalitäten innerhalb des Portals verwiesen. Das Portal ist, wie beschrieben, ein Zusatzmodul und kann über den Market heruntergeladen werden. Dementsprechend konnten diese Funktionen nicht im standardmäßigen Umfang von Axon Ivy gefunden werden. Diese zwei Antworten auf die Forenbeiträge haben somit, durch die Verweise auf das Zusatzmodul und die entsprechende Funktionalität in dem besagten Zusatzmodul, neue Aspekte geliefert und die Punktzahlen verändert. Abgesehen von diesen zwei Ausnahmen haben keine Antworten auf Forenbeiträge die Punktzahlen verändert. Manche Antworten haben auf vorhandenen APIs verwiesen, um die entsprechende Logik programmieren zu können. Dabei haben nur die Antworten für „Skip“ und „Separation of Duties“ bei Bonitasoft und für „Shortest Queue“ und „Skip“ bei Axon Ivy wirklich neue Aspekte geliefert, indem sie auf vorher unbekannte APIs verwiesen haben. Auch die Antwort auf „Authorization“ in Bonitasoft hat einen neuen Aspekt aufgezeigt, dass ein Genehmigungskonzept nur in der kostenpflichtigen Version von Bonitasoft vorhanden ist. Nach diesem Anreiz konnte der Aspekt auch in

der offiziellen Dokumentation gefunden werden [36]. Alle anderen Antworten auf die Forenbeiträge haben die vorher vermutete Einschätzung nur unterstrichen.

Grundsätzlich verfügt Axon Ivy nur über wenig Blogbeiträge, im Vergleich zu Camunda, um auf bereits veröffentlichten Lösungen aufzubauen. Trotz der geringen Anzahl an veröffentlichten Tutorials oder Blogbeiträgen im Vergleich mit Camunda, hat Axon Ivy den mit Abstand besten Forensupport. Insgesamt wurden alle 15 gestellten Fragen beantwortet, was einer Beantwortungsquote von 100% entspricht. Der größte Teil der Fragen wurde innerhalb eines Tages beantwortet und nahezu alle von einem Administrator bzw. direkten Mitarbeiter von Axon Ivy. Auch auf Rückfragen wurde konsequent geantwortet. Dies hat die Arbeit mit Axon Ivy ungemein erleichtert. Auch Bonitasoft hat, wie Axon Ivy, eine deutlich kleinere Community als Camunda, wodurch es weniger bereits veröffentlichte Tutorials und Blogs gibt. Konträr zu Axon Ivy ist der Forensupport von Bonitasoft sehr dürftig. Von den 17 im Rahmen der vorliegenden Arbeit veröffentlichten Forenbeiträgen, wurden nur fünf beantwortet. Dies entspricht einer Beantwortungsquote von ca. 29%. In Verbindung mit der grundsätzlich geringeren Online-Präsenz, im Gegensatz zu Camunda, erschwert dies die Arbeit mit Bonitasoft ungemein. Camunda hat mit Abstand die größte Online-Präsenz und Community aller drei Tools. Demnach konnten viele Fragen durch bereits veröffentlichte Tutorials oder Blogbeiträge beantwortet werden. Demnach mussten deutlich weniger Forenbeiträge im Rahmen der vorliegenden Arbeit veröffentlicht werden als bei den anderen Tools. Insgesamt mussten nur neun Forenbeiträge veröffentlicht werden und nur zwei Beiträge davon wurden nicht beantwortet. Dies entspricht einer Beantwortungsquote von ca. 77%. Der Forensupport ist somit immer noch sehr gut, allerdings nicht auf einer Stufe mit Axon Ivy. Die Forenantworten bei Camunda waren grundsätzlich gut, allerdings wurden auch viele Fragen von anderen Usern und nicht von Administratoren oder direkten Mitarbeitern von Camunda beantwortet. Auch ist ein Administrator, welcher den ursprünglichen Beitrag beantwortet hat, auf eine Rückfrage nicht eingegangen. Grundsätzlich hat vor allem die enorme Online-Präsenz und Community von Camunda am meisten bei der Umsetzung der Patterns geholfen.

### 3.3.3 Erkenntnisse über die Tools ohne direkten Bezug auf die Patterns

Der erste Aspekt ohne direkten Bezug auf die Patterns, der bei Axon Ivy beachtet werden sollte, ist, dass die Workitems der als sog. BPMN Activities gekennzeichneten Tasks bei der Ausführung keiner Ressource zugeteilt werden können und ihnen auch kein Inhalt zugewiesen werden kann. Die BPMN Activities dienen mehr zur Visualisierung bzw. als eine Art Ordner, da die sog. Workflow Activities, welche die wirkliche Arbeit ausführen, als Subprozesse in den BPMN Activities angeordnet werden können [6]. Des Weiteren ist bei der Umsetzung der Patterns aufgefallen, dass der User Dialog immer vorhanden sein muss, damit ein Prozess in Axon Ivy wie gewünscht ausgeführt wird. Auch wenn Axon Ivy den fehlenden User Dialog nur als Warnung und nicht als Error anzeigt, wird ohne diesen User Dialog der „Output“-Tab einer User Task gar nicht erst benutzt bzw. die Werte werden nicht auf die Variablen zugewiesen. Auch hinterlegter Code wird nicht ausgeführt, da ohne User Dialog der gesamte Tab nicht aktiviert bzw. benutzt wird. Grundsätzlich sind häufig Fehler bei der Zuteilung zu Usern und der Belegung von Variablen aufgetreten, wenn ein User Dialog nicht vorhanden war. Ein weiterer Punkt ist, dass das Debuggen in Axon Ivy standardmäßig möglich ist, indem „ivy.log.info();“ verwendet wird. Der gewünschte Inhalt wird dann zur Laufzeit im Tool in dem Tab „Runtime Log“ angezeigt. Ein Fokus des Tools, welcher für die Resource Patterns nicht relevant ist und daher in diesem Kontext nicht genutzt wurde, sind Abwesenheiten und Vertretungen. Axon Ivy verfügt über viele Standardfunktionen und Berechtigungen für Abwesenheiten und Vertretungen. Grundsätzlich kann Code in Axon Ivy einfach hinzugefügt werde, da jeder relevante Tab ein eigenes „Code“-Fenster beinhaltet. Zusätzlich verfügt Axon Ivy für die Programmierung über eine Vorschau an vorhandenen Methoden und jegliche Eingaben werden direkt beim Programmieren geprüft und es wird angezeigt, falls sie nicht existieren. Dies erleichtert die Programmierung ungemein. Auch

anzubringen ist, dass das Portal in den Überprüfungen der Patterns sehr langsam war. Eine neue Seite hat häufig fast 30 Sekunden gebraucht, um zu laden. Die Developer Workflow UI war deutlich schneller und damit ähnlich schnell wie die anderen Tools. Dies könnte auch an der verwendeten Hardware liegen. Des Weiteren kann angemerkt werden, dass die Dokumentation der APIs in Axon Ivy verbesserungswürdig ist. Häufig wird eine Funktion als „Deprecated“ also veraltet betitelt, ist aber trotzdem noch anwendbar im Tool und die angebotene Alternative erfüllt nicht die gleiche Funktionalität. Zum Beispiel ist es möglich, sämtliche Usernamen einer Rolle mit dem Befehl „getAllUsers()“ als eine Liste zu speichern. Dieser Befehl ist aber veraltet und die vorgeschlagene Alternative „users().allPaged()“ liefert die User in einem Axon Ivy internen Datentyp. Es konnte keine alternative Möglichkeit gefunden werden, nur die Usernamen oder IDs aus diesem Datentyp zu extrahieren.

Für Bonitasoft ist der erste Aspekt ohne direkten Bezug auf die Patterns, dass es keinen Standard-Administrator User gibt, wie bei Axon Ivy oder Bonitasoft. Jeder neu erstellte User hat automatisch die Administrator-Berechtigungen, welche in der „Admin“-Weboberfläche eingeschränkt werden müssen, indem der User aus dem Administrator-Profil entfernt wird. Auch ist aufgefallen, dass bei der Erstellung einer neuen Organisation das erste Bereitstellen Probleme bereiten kann. Für das Bereitstellen muss ein User aus der aktiven Organisation in „Authentication“ vorhanden sein. Falls eine neue Organisation erstellt wurde, muss diese aber zuerst aktiviert werden, bevor User aus dieser Organisation verwendet werden können. Demnach muss eine neue Organisation zuerst bereitgestellt werden, wobei noch ein User der alten, noch aktiven Organisation in „Authentication“ eingetragen werden muss. Erst nach diesem Bereitstellen kann die neue Organisation aktiv gesetzt werden und ein User der neuen Organisation in „Authentication“ eingetragen werden. Demnach ist es ratsam, die alte Organisation nicht vor der ersten Bereitstellung der neuen Organisation zu entfernen. Die grundsätzliche Funktionsweise des Actors und dementsprechend des Actor Mappings und der Actor Filter wurden in den Patterns ausführlich erläutert. Allerdings ist es hilfreich anzumerken, dass für die konkrete Zuteilung der Workitems der Actor Filter den Actor überschreibt, welcher in der Human Task hinterlegt wurde, und der Actor aus der Human Task, den Actor aus der Swimlane überschreibt. Auch das Debuggen ist allgemein in Bonitasoft, mithilfe des Befehls „logger.info()“, nur in der Log-Datei unter „C:\BonitaStudioCommunity-2023.2-u0\workspace\.metadata\tomcat.log“ einsehbar. Weitere Aspekte ohne direkten Bezug auf die Patterns, wie die „Swimlane“-Funktionalität und das Hinterlegen eines Managers bei der Usererstellung, mussten in den Erklärungen zu den Patterns bereits detailliert erläutert werden.

Der erste Aspekt ohne direkten Bezug auf die Patterns in Camunda ist das Berechtigungskonzept. Dieses ist, wie in den Untersuchungen der Patterns beschrieben, sehr umfangreich und flexibel, was eine der größten Stärken von Camunda ist. Allerdings müssen daher für sehr viele Elemente zuerst Berechtigungen vergeben werden. Jeder neu erstellte User bzw. jede neu angelegte Gruppe muss immer zuerst per Genehmigung Zugang zu den einzelnen Weboberflächen erhalten. Auch ist wichtig zu beachten, dass Camunda die Berechtigungen für einzelne User nicht entfernt, wenn der User gelöscht wird. Diese müssen manuell gepflegt werden. Darüber hinaus können in der Community Version von Camunda weder abgeschlossene Cases noch abgeschlossene Workitems betrachtet werden. Diese Funktionalitäten sind nur in der Enterprise Version verfügbar [48]. Außerdem gibt es keine Möglichkeit, den aktuellen Status eines Workitems einzusehen, wie in „Distribution by Offer - Single Resource“ beschrieben. Im Cockpit wird zwar unter „State“ ein grüner Haken oder ein rotes „X“ angezeigt, je nachdem ob das Workitem erfolgreich bearbeitet wurde oder es einen Fehler gab, genauere Statusinformationen sind allerdings nicht einsehbar. Die Status aus dem Task Lifecycle [52] können weder nativ noch per Code im Tool eingesehen werden. Es könnte sein, dass sich die Status in der sog. History View einsehen lassen, da diese allerdings nur in der Enterprise Version von Camunda verfügbar ist [48], konnte diese These in der vorliegenden Arbeit nicht überprüft werden. Des Weiteren gibt es im Modeler von Camunda, im

Gegensatz zu den anderen Tools, keine Korrektur oder Autovervollständigung. Dies erschwert das Arbeiten mit Code oder auch Variablen, da ein Rechtschreibfehler unentdeckt bleiben kann bis zur Bereitstellung bzw. Ausführung. Das Debuggen ist in der „Camunda Run“-Windows-Konsole von Camunda möglich durch „System.out.println();“. Auch verfügt Camunda über viele verschiedene APIs, mit einer guten Dokumentation. Camunda ist grundsätzlich eher offen für externe Systeme und verweist bei Funktionen für das Organisationmodell, wie z.B. Hierarchien, Beziehungen oder auch zusätzliche Attribute, auf externe Möglichkeiten wie LDAP (s. digitaler Anhang – Beantwortet & Öffentlich).

Sämtliche Angaben bezüglich der Funktionalitäten aller drei Tools wurden im Rahmen der Untersuchung der Resource Patterns erarbeitet. Demnach basieren die Ergebnisse auf einer Betrachtung der Tools aus der Resource Perspektive und andere Aspekte bzw. Funktionalitäten der Tools werden nicht miteinbezogen.

# 4 Fazit und Ausblick

## 4.1 Fazit der Arbeit

Die zentrale Fragestellung der Arbeit „Wie sind Resource Patterns in den heutigen BPMN basierten WfMS umgesetzt?“ kann durch die Ergebnisse der in der vorliegenden Arbeit durchgeführten Untersuchung beantwortet werden. Auch die folgenden Teilfragestellungen pro Pattern lassen sich durch die Ergebnisse lösen:

- Q1: Kann das Resource Pattern in den heutigen BPMN basierten WfMS abgebildet werden?
- Q2: Wenn ja, wie gut lässt sich das Resource Pattern in den heutigen BPMN basierten WfMS abbilden?

Durch das Bewertungsschema konnten jedem Tool pro Pattern eine Punktezahl von 0 bis 10 zugeordnet werden, wie gut das Tool das Pattern abbilden kann. Die Punktzahl 0 steht dabei dafür, dass das Pattern gar nicht bzw. nur durch individuell programmierten Code umgesetzt werden kann und die Punktzahl 10 dafür, dass es standardmäßig optimal im Tool umgesetzt werden kann. Diese Bewertungen pro Pattern beantworten bereits die Teilfragen Q1 und Q2. Abschließend wurden die Punktzahlen aller Patterns für ein Tool zusammengerechnet, um eine durchschnittliche Gesamtbewertung für ein Tool zu erhalten. Diese durchschnittliche Gesamtbewertung stellt dar, wie gut die Resource Patterns in dem Tool umgesetzt werden können. Die Abbildung 14 fasst die benannten Ergebnisse zusammen und beantwortet somit sowohl die zentrale Fragestellung als auch die Teilfragen pro Pattern durch die einzelnen Zeilen der Tabelle. Axon Ivy erhielt dabei die beste Bewertung mit ca. 5,76 von 10 Punkten. Knapp dahinter lag Camunda mit einer durchschnittlichen Punktzahl von ca. 5,48 von 10. Bonitasoft erhielt mit einem deutlichen Abstand die niedrigste durchschnittliche Punktzahl von ca. 4,65 von 10. Zum Vergleich erhielt YAWL, welches außer Konkurrenz mitbetrachtet wurde, eine Gesamtpunktzahl von ca. 8,14 von 10. An diesen durchschnittlichen Gesamtpunktzahlen kann abgelesen werden, wie gut das jeweilige Tool die Resource Patterns umsetzten kann. Zusätzlich können auch konkrete Prioritäten bei einem WfMS in der Abbildung 14 überprüft werden, durch die Bewertungen der einzelnen Patterns. Hier lassen sich auch die Stärken bzw. Schwächen der Tools bezüglich der Resource Patterns erkennen, welche in dem Kapitel „3.3.1 Vergleich der Tools aufgrund der Untersuchungsergebnisse der Patterns“ ausführlich erläutert wurden. Axon Ivy, welches die höchste Punktzahl erhalten hat, und somit die Resource Patterns am besten abbilden kann, überzeugt vor allem durch die breite Auswahl an Funktionalitäten. Es positioniert sich als Allrounder, allerdings sind dadurch häufig die einzelnen Funktionen nicht so umfassend bzw. tiefgehend wie die der anderen Tools. Axon Ivy ist daher vor allem für Organisationen interessant, welche viele verschiedene Aspekte der Resource Perspektive abdecken wollen, ohne dabei besonders tiefgehende Funktionalitäten in speziellen Bereichen zu benötigen. Bonitasoft konnte mit der geringsten durchschnittlichen Punktzahl die Resource Patterns am schlechtesten abbilden. Vor allem die fehlenden Funktionalitäten tragen zu diesem Ergebnis bei. Trotzdem kann es durch seine Stärken in dem Organisationsmodell und den Zuteilungsmechanismen entsprechende Patterns besser umsetzten als die anderen beiden Tools. Bonitasoft kann somit einer Organisation weiterhelfen, welche den Fokus auf ein komplexes Organisationsmodell samt Zuteilungsmöglichkeiten legt, und beabsichtigt fehlende Funktionalitäten eigenständig umzusetzen. Camunda überzeugt mit der zweitbesten Gesamtpunktzahl vor allem durch das flexible Berechtigungsmanagement und durch die Möglichkeit den Fortschritt in einem Workitem abzuspeichern. Allerdings ist die Umsetzung des Organisationsmodell ausbaufähig. Somit ist Camunda vor allem einer Organisation ratsam, welche komplexe Berechtigungsstrukturen benötigt und beabsichtigt das Organisationsmodell in einem externen System abzubilden. Sollte zusätzlich ein häufiger Austausch von Aufgaben benötigt sein, bei dem der Fortschritt nicht verloren gehen darf, kann Camunda sein volles Potenzial entfalten.

Das Ziel der Arbeit wurde somit erreicht. Durch die Untersuchung und die daraus entstandenen Ergebnisse, wurde betrachtet wie sich die Resource Patterns in drei aktuellen BPMN basierten WfMS umsetzen lassen. Aufgrund dieser Ergebnisse können die Tools miteinander verglichen werden. Es wurden neue Ergebnisse bezüglich der Resource Patterns in modernen WfMS geschaffen, welche Auskunft darüber geben, wie die Resource Perspektive in aktuellen Tools berücksichtigt wird. Zusätzlich wurde ein detailliertes Bewertungsschema erstellt, welches auf beliebige WfMS angewendet werden kann. Auf diese Weise können die Resource Patterns einheitlich in verschiedenen WfMS untersucht und die Tools anschließend verglichen werden. Dieses Bewertungsschema bietet differenziertere Möglichkeiten die Umsetzung der Patterns zu bewerten als das bisher verwendete Schema der Workflow Patterns Initiative.

## 4.2 Ausblick für Resource Patterns und WfMS

In der vorliegenden Arbeit wurden BPMN basierte WfMS mithilfe der Resource Patterns analysiert. Aufgrund der Ergebnisse konnten verschiedene Schwächen der Tools bezüglich der Resource Patterns identifiziert werden. Einzelne Patterns konnten von keinem der Tools umgesetzt werden. Selbst YAWL, welches auf Grundlage der Patterns erstellt wurde, konnte einzelne Patterns nicht abbilden. Dies lässt Freiraum für zwei Interpretationen und somit Ausblicke für die Zukunft: Die Resource Patterns werden weiterentwickelt und ggf. an moderne Anforderungen angepasst oder die Resource Perspektive wird in den WfMS gestärkt.

Die Resource Patterns sind weiterhin relevant und werden in der aktuellen Literatur verwendet. Trotzdem finden sich in der Literatur sowohl Versuche neue Resource Patterns zu entwickeln [57, 66, 72, 100] als auch Kritik, da die Resource Patterns moderne Anforderungen, wie die Möglichkeit mit mehreren Ressourcen an einem Workitem zu arbeiten, nicht ausreichend abbilden [22]. Dementsprechend ist es gut vorstellbar, dass die Resource Patterns erneuert werden und um fehlende oder unterrepräsentierte Aspekte erweitert werden. WfMS bzw. BPMS haben sich bereits weiterentwickelt und umfassen heute weitere Technologien wie Process Mining, Robotic Process Automation (RPA) oder Machine Learning (ML)/Artificial Intelligence (AI) [102]. Somit könnte eine Überarbeitung der Resource Patterns, welche den Effekt der neuen Technologien auf die Resource Perspektive berücksichtigt, hilfreich sein. Auf diese Weise könnten Patterns hinzugefügt werden, welche z.B. AI als Zuordnungsmechanismus für die Ressourcenauswahl heranziehen oder grundsätzlich die Verantwortlichkeiten zwischen Ressourcen beschreiben, die dasselbe Workitem bearbeiten. Auch könnten Patterns, wie „Pre-Do“ und „Early Distribution“, überarbeitet oder entfernt werden. Diese konnten in keinem betrachteten Tool umgesetzt werden, was darauf hinweisen könnte, dass sie nicht praxisrelevant sind oder keinen Mehrwert erbringen.

In Zukunft könnte auch die Resource Perspektive der BPMN basierten WfMS gestärkt werden. Vor allem Patterns wie „Separation of Duties“, „Shortest Queue“ oder auch „Round Robin Allocation“, welche praxisnahe Vorgehensweisen beschreiben, könnten zukünftig nativ vorhanden sein. Dabei könnten sich die Tools an den beispielhaften Implementierungen dieser Patterns in YAWL orientieren. Darüber hinaus legt die bereits beschriebene Entwicklung der WfMS bzw. BPMS nahe, dass sich WfMS bzw. BPMS grundsätzlich weiterhin verändern und im Zuge dessen umfangreicher werden. Bereits heute umfassen sie verschiedene prozessbezogene Technologien, wie RPA, Prozess Mining oder AI. Dies lässt sich auch am Namen des betrachteten Reports erkennen: Digital Process Automation Software. Die verschiedenen Systeme und Technologien werden in einem Produkt gebündelt, um Prozesse effektiver erfassen, verwalten, ausführen und überwachen zu können. Dies könnte auch in Zukunft weiter voranschreiten, sodass WfMS weitere neue Technologien aufnehmen, um umfangreichere Funktionalitäten für das Prozessmanagement zu bieten und sich weiter in Richtung Prozessautomatisierung/-optimierung zu entwickeln. Beispielhaft könnten neue AI Entwicklungen verwendet werden, um Prozesse automatisch zu modellieren und bei Veränderungen selbständig anzupassen. Auch könnte die Ressourcenauswahl optimiert werden, indem

die AI sämtlich ressourcenbezogenen Aspekte, die in den Systemen vorhanden sind, miteinbezieht, wie z.B. die Urlaubsanträge. AI könnte auch die RPA-Technologie unterstützen, sodass diese auch unstrukturierte Aufgaben erledigen könnte, die bisher menschliche Ressourcen erfordern.

In zukünftigen Arbeiten könnte die durchgeführte Untersuchung auf weitere BPMN oder auch nicht BPMN basierte WfMS sowie kommerzielle Tools ausgeweitet werden. Auf diese Weise könnte ein breiterer Überblick über die Umsetzung der Resource Patterns erlangt werden, wodurch eine fundiertere Einschätzung möglich wäre. Eine Arbeit, welche den Fokus auf das Programmieren legt, könnte auch untersuchen, in welchem der drei Tools die zu programmierenden Kriterien am besten umgesetzt werden können. Aufgrund der Bearbeitungsdauer der vorliegenden Arbeit, sind mittlerweile bereits neue Versionen von Axon Ivy und Bonitasoft erschienen. Da auch Camunda mit Camunda 8 eine neue Version aufweist, könnten auch in Zukunft neue Versionen der Tools untersucht werden, um potenzielle Unterschiede in der Umsetzung der Resource Patterns zu identifizieren. Abschließend könnten in zukünftigen Arbeiten auch Kriterien für die Patterns der Control-Flow und der Data Perspektive erarbeitet werden, um das Bewertungsschema auf sämtliche Perspektiven anwenden zu können.

## 5 Literaturverzeichnis


[1] G. Adamo, C. Ghidini, and C. Di Francescomarino, "What is a process model composed of?: A systematic literature review of meta-models in BPM," *Softw Syst Model*, vol. 20, no. 4, pp. 1215–1243, 2021, doi: 10.1007/s10270-020-00847-w.

[2] M. Arias, R. Saavedra, M. R. Marques, J. Munoz-Gama, and M. Sepúlveda, "Human resource allocation in business process management and process mining: A systematic mapping study," *MD*, vol. 56, no. 2, pp. 376–405, 2018, doi: 10.1108/MD-05-2017-0476.

[3] Axon Ivy AG (Hrsg.), *Axon Ivy Designer | DEV Axon Ivy,* 2024. [Online]. Available: https://developer.axonivy.com/download (accessed: Sep. 17 2024).

[4] Axon Ivy AG (Hrsg.), *Axon Ivy Market,* 2024. [Online]. Available: https://market.axonivy.com/ (accessed: Sep. 19 2024).

[5] Axon Ivy AG (Hrsg.), *Axon Ivy Portal — Portal Guide 11.3.1 documentation,* 2024. [Online]. Available: https://market.axonivy.com/portal/11.3/doc/portal-user-guide/axon-ivy-portal/index.html (accessed: Sep. 19 2024).

[6] Axon Ivy AG (Hrsg.), *BPMN Activity Elements — Axon Ivy Platform 11.2 documentation,* 2024. [Online]. Available: https://developer.axonivy.com/doc/11.2/designer-guide/process-modeling/process-elements/bpmn-activity-elements.html (accessed: Jul. 19 2024).

[7] Axon Ivy AG (Hrsg.), *Browsers — Axon Ivy Platform 11.2 documentation,* 2024. [Online]. Available: https://developer.axonivy.com/doc/11.2/designer-guide/ivyscript/browsers.html (accessed: Sep. 17 2024).

[8] Axon Ivy AG (Hrsg.), *Common Tabs — Axon Ivy Platform 11.2 documentation,* 2024. [Online]. Available: https://developer.axonivy.com/doc/11.2/designer-guide/process-modeling/process-elements/common-tabs.html (accessed: Sep. 17 2024).

[9] Axon Ivy AG (Hrsg.), *Customize Task Delegation List — Portal Guide 11.2.1 documentation,* 2024. [Online]. Available: https://market.axonivy.com/market-cache/portal/portal-guide/11.2.1/portal-developer-guide/customization/task-delegation.html (accessed: Jul. 8 2024).

[10] Axon Ivy AG (Hrsg.), *Developer Workflow UI — Axon Ivy Platform 11.2 documentation,* 2024. [Online]. Available: https://developer.axonivy.com/doc/11.2/designer-guide/how-to/workflow/dev-workflow-ui.html (accessed: Sep. 17 2024).

[11] Axon Ivy AG (Hrsg.), *IRole getProperty - Documentation: Axon Ivy Public API (Version 11.2.2.2404020001),* 2024. [Online]. Available: https://developer.axonivy.com/doc/11.2/public-api/ch/ivyteam/ivy/security/IRole.html#getProperty(java.lang.String) (accessed: Jun. 12 2024).

[12] Axon Ivy AG (Hrsg.), *ISecurityDescriptor - Documentation: Axon Ivy Public API (Version 11.2.2.2404020001),* 2024. [Online]. Available: https://developer.axonivy.com/doc/11.2/public-api/ch/ivyteam/ivy/security/ISecurityDescriptor.html (accessed: Jun. 10 2024).

[13] Axon Ivy AG (Hrsg.), *Leading Edge Versions 11.2.1 | DEV Axon Ivy,* 2024. [Online]. Available: https://developer.axonivy.com/download/archive/11 (accessed: Sep. 9 2024).

[14] Axon Ivy AG (Hrsg.), *Process Editor — Axon Ivy Platform 11.2 documentation,* 2024. [Online]. Available: https://developer.axonivy.com/doc/11.2/designer-guide/process-modeling/process-modeling/process-editor.html (accessed: Sep. 17 2024).

[15] Axon Ivy AG (Hrsg.), *Roles and Users — Axon Ivy Platform 11.2 documentation,* 2024. [Online]. Available: https://developer.axonivy.com/doc/11.2/designer-guide/configuration/roles-users.html (accessed: Jun. 10 2024).

[16] Axon Ivy AG (Hrsg.), *Simulation — Axon Ivy Platform 11.2 documentation,* 2024. [Online]. Available: https://developer.axonivy.com/doc/11.2/designer-guide/process-modeling/simulation/simulation.html (accessed: Sep. 17 2024).

[17] Axon Ivy AG (Hrsg.), *Swimlanes — Axon Ivy Platform 11.2 documentation,* 2024. [Online]. Available: https://developer.axonivy.com/doc/11.2/designer-guide/process-modeling/process-modeling/swimlanes.html (accessed: Jun. 5 2024).

[18] Axon Ivy AG (Hrsg.), *TaskState - Documentation: Axon Ivy Public API (Version 11.2.2.2404020001),* 2024. [Online]. Available: https://developer.axonivy.com/doc/11.2/public-api/ch/ivyteam/ivy/workflow/TaskState.html (accessed: Sep. 17 2024).

[19] Axon Ivy AG (Hrsg.), *The Forrester Wave™ | Axon Ivy,* 2024. [Online]. Available: https://www.axonivy.com/downloads/forrester-wave-23 (accessed: Sep. 9 2024).

[20] Axon Ivy AG (Hrsg.), *User Task — Axon Ivy Platform 11.2 documentation,* 2024. [Online]. Available: https://developer.axonivy.com/doc/11.2/designer-guide/process-modeling/process-elements/user-task.html (accessed: Sep. 17 2024).

[21] Berliner BPM-Offensive, *BPMN Poster: BPMN 2.0 - Business Process Model and Notation,* 2011. [Online]. Available: https://www.signavio.com/de/news/das-bpmn-2-0-poster-jetzt-kostenlos-zuschicken-lassen/ (accessed: Aug. 30 2024).

[22] R. Bidar, A. ter Hofstede, R. Sindhgatta, and C. Ouyang, "Preference-Based Resource and Task Allocation in Business Process Automation," in *Lecture Notes in Computer Science, On the Move to Meaningful Internet Systems: OTM 2019 Conferences*, Panetto and Birukou, Eds., 1st ed., [Place of publication not identified]: Springer International Publishing, 2019, pp. 404–421, doi: 10.1007/978-3-030-33246-4_26.

[23] P. Bocciarelli, A. D'Ambrogio, A. Giglio, and E. Paglia, "Modeling Resources to Simulate Business Process Reliability," *ACM Trans. Model. Comput. Simul.*, vol. 30, no. 3, pp. 1–25, 2020, doi: 10.1145/3381453.

[24] Bonitasoft S.A. (Hrsg.), *Bonita process automation software in Forrester DPA Wave Q4 2023,* 2023. [Online]. Available: https://www.bonitasoft.com/news/bonita-process-automation-software-in-forrester-dpa-wave-q4-2023 (accessed: Sep. 9 2024).

[25] Bonitasoft S.A. (Hrsg.), *Actor filters | Bonita Documentation,* 2024. [Online]. Available: https://documentation.bonitasoft.com/bonita/2023.2/process/actor-filtering (accessed: Sep. 18 2024).

[26] Bonitasoft S.A. (Hrsg.), *Actors of processes | Bonita Documentation,* 2024. [Online]. Available: https://documentation.bonitasoft.com/bonita/2023.2/process/actors (accessed: Sep. 18 2024).

[27] Bonitasoft S.A. (Hrsg.), *Bonita components | Bonita Documentation,* 2024. [Online]. Available: https://documentation.bonitasoft.com/bonita/2023.2/bonita-overview/bonita-bpm-overview (accessed: Sep. 17 2024).

[28] Bonitasoft S.A. (Hrsg.), *Bonita Studio | Bonita Documentation,* 2024. [Online]. Available: https://documentation.bonitasoft.com/bonita/2023.2/bonita-overview/bonita-studio (accessed: Sep. 18 2024).

[29] Bonitasoft S.A. (Hrsg.), *BPM execution states | Bonita Documentation,* 2024. [Online]. Available: https://documentation.bonitasoft.com/bonita/2023.2/runtime/execution-sequence-states-and-transactions (accessed: Jul. 5 2024).

[30] Bonitasoft S.A. (Hrsg.), *Contracts and contexts* | *Bonita Documentation,* 2024. [Online]. Available: https://documentation.bonitasoft.com/bonita/2023.2/data/contracts-and-contexts (accessed: Sep. 18 2024).

[31] Bonitasoft S.A. (Hrsg.), *Forms* | *Bonita Documentation,* 2024. [Online]. Available: https://documentation.bonitasoft.com/bonita/2023.2/pages-and-forms/forms (accessed: Sep. 18 2024).

[32] Bonitasoft S.A. (Hrsg.), *Forms development* | *Bonita Documentation,* 2024. [Online]. Available: https://documentation.bonitasoft.com/bonita/2023.2/pages-and-forms/forms-development (accessed: Sep. 17 2024).

[33] Bonitasoft S.A. (Hrsg.), *Operations to update variables value* | *Bonita Documentation,* 2024. [Online]. Available: https://documentation.bonitasoft.com/bonita/2023.2/process/operations (accessed: Sep. 18 2024).

[34] Bonitasoft S.A. (Hrsg.), *Organization overview* | *Bonita Documentation,* 2024. [Online]. Available: https://documentation.bonitasoft.com/bonita/2023.2/identity/organization-overview (accessed: Sep. 18 2024).

[35] Bonitasoft S.A. (Hrsg.), *Pools and lanes* | *Bonita Documentation,* 2024. [Online]. Available: https://documentation.bonitasoft.com/bonita/2023.2/process/pools-and-lanes (accessed: Sep. 18 2024).

[36] Bonitasoft S.A. (Hrsg.), *Profiles overview* | *Bonita Documentation,* 2024. [Online]. Available: https://documentation.bonitasoft.com/bonita/2023.2/identity/profiles-overview (accessed: Jun. 10 2024).

[37] Bonitasoft S.A. (Hrsg.), *Specify data in a process definition* | *Bonita Documentation,* 2024. [Online]. Available: https://documentation.bonitasoft.com/bonita/2023.2/data/specify-data-in-a-process-definition (accessed: Sep. 18 2024).

[38] Bonitasoft S.A. (Hrsg.), *UI Designer overview* | *Bonita Documentation,* 2024. [Online]. Available: https://documentation.bonitasoft.com/bonita/2023.2/bonita-overview/ui-designer-overview (accessed: Sep. 18 2024).

[39] Bonitasoft S.A. (Hrsg.), *What is Bonita?* | *Bonita Documentation,* 2024. [Online]. Available: https://documentation.bonitasoft.com/bonita/2023.2/bonita-overview/what-is-bonita-index (accessed: Sep. 9 2024).

[40] Camunda GmbH (Hrsg.), *Get Authorizations - Documentation,* 2023. [Online]. Available: https://docs.camunda.org/manual/7.15/reference/rest/authorization/get-query/ (accessed: Jun. 10 2024).

[41] Camunda GmbH (Hrsg.), *Get Tasks (Historic) - Documentation,* 2023. [Online]. Available: https://docs.camunda.org/manual/7.15/reference/rest/history/task/get-task-query/ (accessed: Jun. 27 2024).

[42] Camunda GmbH (Hrsg.), *Admin - Documentation,* 2024. [Online]. Available: https://docs.camunda.org/manual/7.21/webapps/admin/ (accessed: Sep. 18 2024).

[43] Camunda GmbH (Hrsg.), *Camunda 7 documentation* | *docs.camunda.org,* 2024. [Online]. Available: https://docs.camunda.org/manual/7.21/ (accessed: Sep. 9 2024).

[44] Camunda GmbH (Hrsg.), *Camunda 8 Self-Managed* | *Camunda 8 Docs,* 2024. [Online]. Available: https://docs.camunda.io/docs/self-managed/about-self-managed/ (accessed: Sep. 9 2024).

[45] Camunda GmbH (Hrsg.), *Cockpit - Documentation,* 2024. [Online]. Available: https://docs.camunda.org/manual/7.21/webapps/cockpit/ (accessed: Sep. 18 2024).

[46] Camunda GmbH (Hrsg.), *Delegation Code - Documentation,* 2024. [Online]. Available: https://docs.camunda.org/manual/7.21/user-guide/process-engine/delegation-code/ (accessed: Sep. 18 2024).

[47] Camunda GmbH (Hrsg.), *Download Camunda 7,* 2024. [Online]. Available: https://camunda.com/de/download/platform-7/ (accessed: Sep. 18 2024).

[48] Camunda GmbH (Hrsg.), *History in Cockpit - Documentation,* 2024. [Online]. Available: https://docs.camunda.org/manual/7.21/webapps/cockpit/bpmn/process-history-views/ (accessed: Jul. 19 2024).

[49] Camunda GmbH (Hrsg.), *Identity Service - Documentation,* 2024. [Online]. Available: https://docs.camunda.org/manual/7.21/user-guide/process-engine/identity-service/ (accessed: Jun. 10 2024).

[50] Camunda GmbH (Hrsg.), *[report] The Forrester Wave™ DPA 2023* | *Camunda,* 2024. [Online]. Available: https://page.camunda.com/wp-forrester-dpa-wave (accessed: Sep. 9 2024).

[51] Camunda GmbH (Hrsg.), *Scripting - Documentation,* 2024. [Online]. Available: https://docs.camunda.org/manual/7.21/user-guide/process-engine/scripting/ (accessed: Sep. 18 2024).

[52] Camunda GmbH (Hrsg.), *Task Lifecycle - Documentation,* 2024. [Online]. Available: https://docs.camunda.org/manual/7.21/webapps/tasklist/task-lifecycle/ (accessed: Jul. 5 2024).

[53] Camunda GmbH (Hrsg.), *Tasklist - Documentation,* 2024. [Online]. Available: https://docs.camunda.org/manual/7.21/webapps/tasklist/ (accessed: Sep. 18 2024).

[54] Camunda GmbH (Hrsg.), *User Task Forms - Documentation,* 2024. [Online]. Available: https://docs.camunda.org/manual/7.21/user-guide/task-forms/ (accessed: Sep. 17 2024).

[55] Camunda GmbH (Hrsg.), *Working with Tasklist - Documentation,* 2024. [Online]. Available: https://docs.camunda.org/manual/7.21/webapps/tasklist/working-with-tasklist/ (accessed: Sep. 17 2024).

[56] R. Carvalho, H. Mili, A. Boubaker, J. Gonzalez-Huerta, and S. Ringuette, *On the Analysis of CMMN Expressiveness: Revisiting Workflow Patterns,* 2016. [Online]. Available: https://www.win.tue.nl/~rmedeiro/papers/rapport-latece.pdf (accessed: Aug. 23 2024).

[57] R. Da Ferreira Silva *et al.,* "Workflows Community Summit: Advancing the State-of-the-art of Scientific Workflows Management Systems Research and Development," 2021. [Online]. Available: http://arxiv.org/pdf/2106.05177##, doi: 10.5281/zenodo.4915801.

[58] N. Deehan, *Camunda Platform 8 for Camunda Platform 7 Users – What You Need to Know,* 2022. [Online]. Available: https://camunda.com/blog/2022/04/camunda-platform-8-for-camunda-platform-7-users-what-you-need-to-know/ (accessed: Sep. 9 2024).

[59] F. Durán, C. Rocha, and G. Salaün, "Analysis of the Runtime Resource Provisioning of BPMN Processes Using Maude," in *Lecture Notes in Computer Science, REWRITING LOGIC AND ITS APPLICATIONS: 13th international workshop, wrla*, S. Escobar and N. Martí-Oliet, Eds., [S.l.]: Springer, 2021, pp. 38–56, doi: 10.1007/978-3-030-63595-4_3.

[60] V. Dwivedi, V. Pattanaik, V. Deval, A. Dixit, A. Norta, and D. Draheim, "Legally Enforceable Smart-Contract Languages: A Systematic Literature Review," *ACM Comput. Surv.*, vol. 54, no. 5, pp. 1–34, 2021, doi: 10.1145/3453475.

[61] Forrester Research, Inc., *Research: The Forrester Wave™,* 2024. [Online]. Available: https://www.forrester.com/research/ (accessed: Sep. 9 2024).

[62] Forrester Research, Inc., *The Forrester Wave Methodology,* 2024. [Online]. Available: https://www.forrester.com/policies/forrester-wave-methodology/ (accessed: Sep. 9 2024).

[63] I. Garfatta, K. Klai, M. Graiet, and W. Gaaloul, "Formal Modelling and Verification of Cloud Resource Allocation in Business Processes," in *Lecture Notes in Computer Science, On the Move to Meaningful Internet Systems. OTM 2018 Conferences*, Hofmann and Panetto, Eds., Cham: Springer International Publishing, 2018, pp. 552–567, doi: 10.1007/978-3-030-02610-3_31.

[64] S. Ghilardi, A. Gianola, M. Montali, and A. Rivkin, "Petri net-based object-centric processes with read-only data," *Information Systems*, vol. 107, p. 102011, 2022, doi: 10.1016/j.is.2022.102011.

[65] A. Gianola, *Verification of Data-Aware Processes Via Satisfiability Modulo Theories,* 1st ed. Cham: Springer, 2023, doi: 10.1007/978-3-031-42746-6.

[66] K. Goel, T. Fehrer, M. Röglinger, and M. T. Wynn, "Not Here, But There: Human Resource Allocation Patterns," in *Lecture Notes in Computer Science, BUSINESS PROCESS MANAGEMENT: 21st international conference, bpm*, C. Di Francescomarino, A. Burattin, C. Janiesch, and S. Sadiq, Eds., [S.l.]: SPRINGER INTERNATIONAL PU, 2023, pp. 377–394, doi: 10.1007/978-3-031-41620-0_22.

[67] K. Grunert, J. Joderi Shoferi, K. Rohwer, E. Pankovska, and L. Gold, "Architecture of decentralized Process Management Systems," in *Lecture Notes in Computer Science, Business Process Management: 20th International Conference, BPM 2022, Münster, Germany, September 11-16, 2022, Proceedings*, C. Di Ciccio, Ed., Cham: Springer, uuuu-uuuu, pp. 436–452, doi: 10.1007/978-3-031-16103-2_28.

[68] E. Hachicha, N. Assy, W. Gaaloul, and J. Mendling, "A Configurable Resource Allocation for Multi-tenant Process Development in the Cloud," in *Lecture notes in computer science. SL 3, Information systems and applications*, vol. 9694, *Advanced information systems engineering: Proceedings*, S. Nurcan, P. Soffer, M. Bajec, and J. Eder, Eds., Switzerland: Springer, 2016, pp. 558–574, doi: 10.1007/978-3-319-39696-5_34.

[69] A. Harris, "Integrating Business Process Management To Model Context In Healthcare: A Case Study Using Perioperative Processes," Master Thesis, Faculty of Engineering, Université d'Ottawa / University of Ottawa, Ottawa, 2016. [Online]. Available: https://ruor.uottawa.ca/handle/10393/34106

[70] Hochschulbibliothekszentrum des Landes Nordrhein-Westfalen (hbz), *Datenbanken & Zeitschriften,* 2024. [Online]. Available: https://h-brs.digibib.net/eres (accessed: Aug. 23 2024).

[71] A. H. M. Hofstede, W. M. P. Aalst, M. Adams, and N. Russell, *Modern Business Process Automation: YAWL and its Support Environment*. Berlin, Heidelberg: Springer, 2010, doi: 10.1007/978-3-642-03121-2.

[72] M. Ilyas, "Hybrid process modelling languages," Master Thesis, 2020. Accessed: Aug. 20 2024. [Online]. Available: https://documentserver.uhasselt.be/bitstream/1942/32170/1/69a9da8c-47e1-45de-a12b-f3d3089f8491.pdf

[73] E. M. L. Jonkmans, "Supporting job quality in manufacturing from a run-time process management perspective," Master Thesis, Industrial Engineering & Innovation Sciences department, Eindhoven University of Technology, Eindhoven, 2020. Accessed: Aug. 23 2024. [Online]. Available: https://pure.tue.nl/ws/portalfiles/portal/172236491/jonkmans_eva.pdf

[74] F. Kossak *et al.,* "A Layered Approach for Actor Modelling," in *Hagenberg Business Process Modelling Method*, F. Kossak et al., Eds., 1st ed., Cham: Springer International Publishing, 2016, pp. 63–84, doi: 10.1007/978-3-319-30496-0_3.

[75] F. Kossak *et al.,* "A Typed Approach to User Interaction Modelling," in *Hagenberg Business Process Modelling Method*, F. Kossak et al., Eds., 1st ed., Cham: Springer International Publishing, 2016, pp. 85–116, doi: 10.1007/978-3-319-30496-0_4.

[76] M. Krämer, "A Microservice Architecture for the Processing of Large Geospatial Data in the Cloud," Dissertation, Fachbereich Informatik, Technischen Universität Darmstadt, Darmstadt, 2017. Accessed: Aug. 23 2024. [Online]. Available: https://diglib.eg.org/handle/10.2312/2632114

[77] J. Krogstie, "Organizational Value of Business Process Modeling," in *Quality in Business Process Modeling*, J. Krogstie, Ed., Cham: Springer International Publishing, 2016, pp. 187–226, doi: 10.1007/978-3-319-42512-2_5.

[78] T. Küster, *Eine Methodik zur Entwicklung von Multiagenten-Systemen auf Basis von Geschäftsprozess-Modellen,* 2017. [Online]. Available: https://api-depositonce.tu-berlin.de/server/api/core/bitstreams/59f30bb8-cb94-4cd5-bae6-513e2f932ee0/content##, doi: 10.14279/depositonce-6549.

[79] R. Laue, A. Koschmider, and D. Fahland, *Prozessmanagement und Process-Mining: Grundlagen*. München, Berlin: De Gruyter Oldenbourg; Walter de Gruyter GmbH, 2021, doi: 10.1515/9783110500165.

[80] C. Le Clair, G. O'Donnell, R. Taylor-Huot, L. S. Sung, A. Lynch, and K. Hartig, *The Forrester Wave™: Digital Process AutomationSoftware, Q4 2023: The 15 Providers That Matter Most And How They Stack Up,* 2023. [Online]. Available: https://page.camunda.com/wp-forrester-dpa-wave (accessed: Sep. 9 2024).

[81] L. Lehmann, "AW_ MI-Seminar Nachfrage Bib-Discover", E-Mail, 10.05.2023.

[82] Z. Luo, Y. Li, R. Fu, and J. Yin, "Don't Fire Me, a Kernel Autoregressive Hybrid Model for Optimal Layoff Plan," in *BigData Congress 2016: 2016 IEEE International Congress on Big Data : proceedings : 27 June-2 July 2016, San Francisco, California, USA*, San Francisco, CA, USA, 2016, pp. 470–477, doi: 10.1109/BigDataCongress.2016.72.

[83] N. A. Mulyar, *Pattern-based Evaluation of Oracle-BPEL (v.10.1.2),* 2005. [Online]. Available: http://www.workflowpatterns.com/vendors/documentation/BPM-05-24.pdf (accessed: Aug. 19 2024).

[84] N. Russell, A.H.M. ter Hofstede, D. Edmond, and W.M.P. van der Aalst., *Workflow Resource Patterns: BETA Working Paper Series,* 2004. [Online]. Available: http://www.workflowpatterns.com/documentation/documents/Resource%20Patterns%20BETA%20TR.pdf (accessed: Jun. 4 2024).

[85] Object Management Group, Inc. (OMG), *Business Process Model and Notation (BPMN): Version 2.0.2,* 2013. [Online]. Available: https://www.omg.org/spec/BPMN/2.0.2/PDF (accessed: Aug. 30 2024).

[86] M. Pesic and W. M. P. van der Aalst, "Modelling work distribution mechanisms using Colored Petri Nets," *Int J Softw Tools Technol Transfer*, vol. 9, 3-4, pp. 327–352, 2007, doi: 10.1007/s10009-007-0036-z.

[87] L. Pufahl, "Modeling and executing batch activities in business processes," Dissertation, Digital-Engineering-Fakultät des Hasso-Plattner-Instituts, Universität Potsdam, Potsdam, 2018. Accessed: Aug. 23 2024. [Online]. Available: https://publishup.uni-potsdam.de/opus4-ubp/frontdoor/deliver/index/docId/40801/file/pufahl_diss.pdf

[88] L. Pufahl and M. Weske, "Batch activity: enhancing business process modeling and enactment with batch processing," *Computing*, vol. 101, no. 12, pp. 1909–1933, 2019, doi: 10.1007/s00607-019-00717-4.

[89] Red Hat, Inc. (Hrsg.), *jBPM - Open Source Business Automation Toolkit - Download: Latest final version: 7.74.1.Final (Release 20.06.2023),* 2023. [Online]. Available: https://www.jbpm.org/download/community.html (accessed: Sep. 9 2024).

[90] J. G. Represa *et al.,* "Investigation of Microservice-Based Workflow Management Solutions for Industrial Automation," *Applied Sciences*, vol. 13, no. 3, p. 1835, 2023, doi: 10.3390/app13031835.

[91] C. Rieger, "Interoperability of BPMN and MAML for Model-Driven Development of Business Apps," in *Lecture Notes in Business Information Processing*, vol. 319, *Business Modeling and Software Design: 8th International Symposium, BMSD 2018, Vienna, Austria, July 2-4, 2018, Proceedings*, B. Šiškov, Ed., Cham, Switzerland: Springer, 2018, pp. 149–166, doi: 10.1007/978-3-319-94214-8_10.

[92] N. Russell, "Foundations of process-aware information systems," Dissertation, Queensland University of Technology, Brisbane, Australien, 2007. Accessed: Aug. 23 2024. [Online]. Available: https://eprints.qut.edu.au/16592

[93] N. Russell, W. van der Aalst, and A. ter Hofstede, *Workflow patterns: The definitive guide*. Cambridge, Massachusetts: MIT Press, 2016. [Online]. Available: https://ebookcentral.proquest.com/lib/kxp/detail.action?docID=5891216

[94] N. Russell, W. M. van der Aalst, A. ter Hofstede, and P. Wohed, *On the Suitability of UML 2.0 Activity Diagrams for Business Process Modelling,* 2006. [Online]. Available: http://www.workflowpatterns.com/documentation/documents/UMLEval-APCCM.pdf (accessed: Aug. 19 2024).

[95] N. Russell, W. M. P. van der Aalst, A. H. M. ter Hofstede, and D. Edmond, "Workflow Resource Patterns: Identification, Representation and Tool Support," in *Advanced Information Systems Engineering: CAiSE 2005. Lecture Notes in Computer Science, vol 3520. Springer, Berlin, Heidelberg.*, 2005, pp. 216–232, doi: 10.1007/11431855_16.

[96] S. Schönig, C. Cabanillas, C. Di Ciccio, S. Jablonski, and J. Mendling, "Mining team compositions for collaborative work in business processes," *Softw Syst Model*, vol. 17, no. 2, pp. 675–693, 2016, doi: 10.1007/s10270-016-0567-4.

[97] N. Schützenmeier, S. Jablonski, and S. Schönig, "Comparing Process Models Beyond Structural Equivalence," in *Lecture Notes in Business Information Processing, Advanced Information Systems Engineering Workshops: CAiSE 2024 International Workshops, Limassol, Cyprus, June 3-7, 2024, Proceedings*, J. P. A. Almeida, C. Di Ciccio, and C. Kalloniatis, Eds., Cham: Springer International Publishing AG, 2024, pp. 291–306, doi: 10.1007/978-3-031-61003-5_25.

[98] Slashdot Media (Hrsg.), *OpenWFE - Download,* 25.04.2013. [Online]. Available: https://sourceforge.net/projects/openwfe/ (accessed: Sep. 9 2024).

[99] Slashdot Media (Hrsg.), *Shark - Java Open Source XPDL Workflow - Download,* 23.11.2018. [Online]. Available: https://sourceforge.net/projects/sharkwf/ (accessed: Sep. 9 2024).

[100] Steven Mertens, Frederik Gailly, and Geert Poels, "Towards a decision-aware declarative process modeling language for knowledge-intensive processes," *Expert Systems with Applications*, vol. 87, pp. 316–334, 2017, doi: 10.1016/j.eswa.2017.06.024.

[101] L. J. R. Stroppi, O. Chiotti, and P. D. Villarreal, "Defining the resource perspective in the development of processes-aware information systems," *Information and Software Technology*, vol. 59, pp. 86–108, 2015, doi: 10.1016/j.infsof.2014.10.006.

[102] M. Szelągowski and A. Lupeikiene, "Business Process Management Systems: Evolution and Development Trends," *Informatica*, pp. 579–595, 2020, doi: 10.15388/20-INFOR429.

[103] The YAWL Foundation (Hrsg.), *YAWL Technical Manual 5.0,* 2022. [Online]. Available: https://yawlfoundation.github.io/assets/files/YAWLTechnicalManual5.0.pdf (accessed: Sep. 23 2024).

[104] The YAWL Foundation (Hrsg.), *Download YAWL,* 2024. [Online]. Available: https://yawlfoundation.github.io/page6.html (accessed: Sep. 23 2024).

[105] W. van der Aalst and A. ter Hofstede, "YAWL: Yet Another Workflow Language," *Information Systems*, vol. 30, no. 4, pp. 245–275, 2005, doi: 10.1016/j.is.2004.02.002.

[106] W. M. P. van der Aalst, M. Dumas, A. H. M. ter Hofstede, and P. Wohed, *Pattern-Based Analysis of BPML (and WSCI),* 2002. [Online]. Available: http://www.workflowpatterns.com/documentation/documents/qut_bpml_rep.pdf (accessed: Sep. 23 2024).

[107] W. M. P. van der Aalst and A. H. M. ter Hofstede, "Workflow patterns put into context," *Softw Syst Model*, vol. 11, no. 3, pp. 319–323, 2012, doi: 10.1007/s10270-012-0233-4.

[108] D. van Nuffel and M. D. Backer, "Multi-abstraction layered business process modeling," *Computers in Industry*, vol. 63, no. 2, pp. 131–147, 2012, doi: 10.1016/j.compind.2011.12.001.

[109] M. Weske, *Business process management: Concepts, languages, architectures*. Berlin, Germany: Springer, 2024, doi: 10.1007/978-3-662-69518-0.

[110] P. Wohed, N. Russell, A. H. ter Hofstede, B. Andersson, and W. M. van der Aalst, "Patterns-based evaluation of open source BPM systems: The cases of jBPM, OpenWFE, and Enhydra Shark," *Information and Software Technology*, vol. 51, no. 8, pp. 1187–1216, 2009, doi: 10.1016/j.infsof.2009.02.002.

[111] P. Wohed, W. M. P. van der Aalst, M. Dumas, A. H. M. ter Hofstede, and N. Russell, "On the Suitability of BPMN for Business Process Modelling," in *Lecture Notes in Computer Science, Business process management: 4th international conference, BPM 2006, Vienna, Austria, September 5-7, 2006 : proceedings*, D. Hutchison et al., Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 161–176, doi: 10.1007/11841760_12.

[112] Workflow Management Coalition (Hrsg.), *Workflow Management Coalition: Terminology & Glossary: Document Number WFMC-TC-1011,* 1999. [Online]. Available: https://healthcareworkflow.files.wordpress.com/2008/10/wfmc-tc-1011_term_glossary_v3.pdf (accessed: Sep. 16 2024).

[113] Workflow Patterns Initiative (Hrsg.), *Workflow Patterns | Documentation,* 2023. [Online]. Available: http://www.workflowpatterns.com/documentation/ (accessed: Aug. 23 2024).

[114] Workflow Patterns Initiative (Hrsg.), *Workflow Patterns | Workflow Resource Patterns,* 2023. [Online]. Available: http://www.workflowpatterns.com/patterns/resource/ (accessed: Sep. 16 2024).

[115] Workflow Patterns Initiative (Hrsg.), *Workflow Patterns Home Page,* 2023. [Online]. Available: http://www.workflowpatterns.com/ (accessed: Sep. 16 2024).