Eskalations-Typen

- Eskalation nach Bearbeitungsfrist ("Deadline")
- Eskalation zur Workflow-Aktion ("Action")
- Eskalation zur Zahlungsbedingung ("PPC")

Eskalation nach Bearbeitungsfrist ("Deadline")

Jede Rechnung sollte innerhalb einer definierten Bearbeitungsfrist gebucht werden. Sobald diese Frist überschritten wird versendet das System eine Eskalations-Email an den aktuell sperrenden Benutzer.

Die Rechnungs-Mappe muss im Umlauf sein. Wenn die Mappe bei einer technischen Aktion für die Buchungs-Schnittstelle oder der Archivierung liegt muss ebenfalls keine Eskalation erfolgen. In der Regel können alle technischen Aktionen ignoriert werden. Die hinterlegte Prüfung kann beliebig geändert order erweitert werden. Zum Beispiel können Rechnungen die auf Wiedervorlage liegen von der Eskalation ausgeschlossen werden.

Die Prüfung aus Eskalation wird in der Funktion "checkEscalation()" durchgeführt. Die Funktion wird pro DocFile-Objekt - also pro Rechnung aufgerufen.

```
switch( this.EscalationType ){
    case "Deadline":
        if( df.ActionStatus==="WaitSplits" || df.ActionStatus==="TechActionJob" || df.ActionStatus==="TechActionReceive" ){
            this.Log.info("["+id+"] checkEscalation() .. df.ActionStatus("+df.ActionStatus+") .. return false");
            return false;
        }

        if( this.Template==="Invoice" ){
            var wfStart = df.WorkflowStart;
            if( wfStart instanceof Date ){
                var wfStartDateStr  = util.convertDateToString(wfStart, "dd.mm.yyyy");
                var wfStartPlusX    = context.addTimeInterval(wfStart, 10, "days", true);
                var wfStartPlusXStr = util.convertDateToString(wfStartPlusX, "dd.mm.yyyy");
                if( wfStartPlusX < new Date() ){
                    this.Log.info("["+id+"] checkEscalation() .. context.addTimeInterval("+wfStart+", 10, \"days\", true)=="+wfStartPlusXStr+" < new Date() "
                                + ".. return true");
                    return true;
                }
                this.Log.info("["+id+"] checkEscalation() .. context.addTimeInterval("+wfStart+", 10, \"days\", true)=="+wfStartPlusXStr+" < new Date() .. return true");
                return false;
            }
            else{
                df.WorkflowStart = new Date();
                if( df.sync() ){
                    this.Log.info("["+id+"] checkEscalation() .. set df.WorkflowStart := new Date() .. return false");
                    return false;
                }
                else{
                    this.Log.info("["+id+"] checkEscalation() .. could not set set df.WorkflowStart: "+df.getLastError()+" .. return false");
                    return false;
                }
            }
        }
```

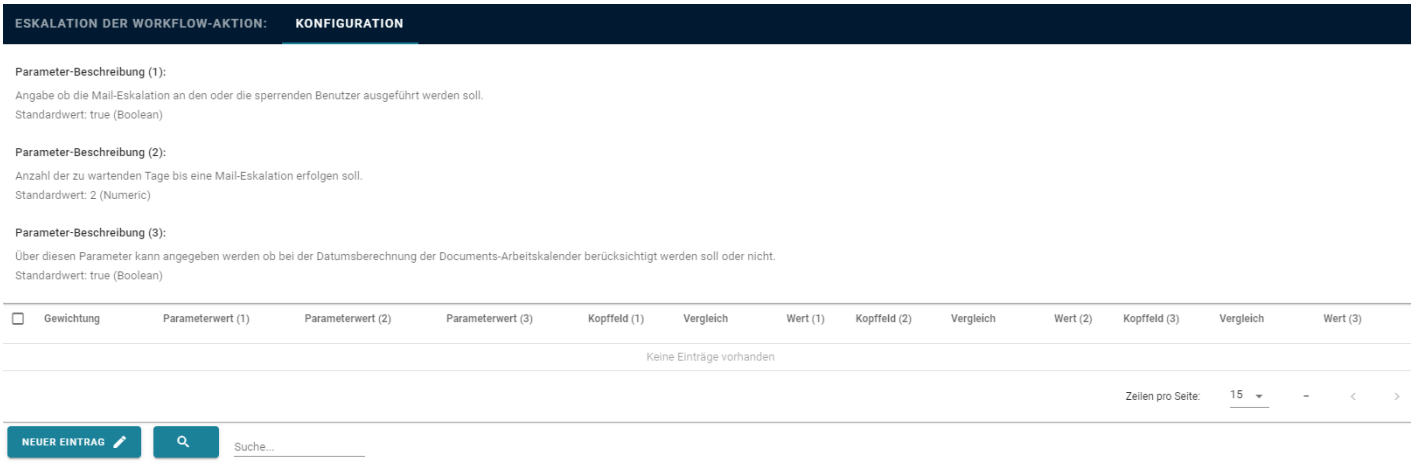
Für die Eskalations-Prüfung wird im Standard das Datumsfeld "WorkflowStart" verwendet, welches direkt bei der Erstellung der Mappe auf einen aktuellen Zeitstempel gesetzt wird. Es kann aber jedes beliebige Datum aber auch jede anders geartete Prüfung ausgeführt werden. Auf das Workflow-Start-Datum werden standardmäßig 10 Arbeitstage aufaddiert. Wenn das berechnete Datum in der Vergangenheit liegt, soll die Rechnung eskalieren ("return true").

Im Anschluss werden die zu informierenden Benutzer über die User-Exit-Funktion "getEscalationUserArray()" ermittelt. Für jeden ermittelten Benutzer wird eine HTML-Datei erstellt und für jede Rechnung wird ein Eintrag hinzugefügt. Am Ende werden die HTML-Tags geschlossen und die Mails werden versendet.

Die Eskalation kann über das Skript "Invoice_Job_EscalatePPC" gesteuert werden.

Eskalation zur Workflow-Aktion ("Action")

Anders als die Eskalation "Deadline" bezieht sich diese Eskalation nicht auf den gesamten. Beim Eskalations-Typen "Action" erfolgt eine Eskalation, wenn eine Rechnung eine definierte Anzahl an Tagen in einer Workflow-Aktion liegt. Die Prüfung auf eine Eskalation erfolgt durch die User-Exit-Funktion "checkEscalation()". Die Funktion verwendet in der Standardauslieferung den Parameter "Escalation_WorkflowAction".



Über den Parameter können 3 Werte angegeben werden:

1. Der erste Parameter-Wert gibt an, ob eine Eskalation via Mail überhaupt stattfinden soll. Der Standardwert für den Parameter ist "true". Die Eskalation kann generell oder abhängig von Feldwerten deaktiviert werden.
2. Über den zweiten Parameter-Wert kann definiert werden, nach wie viel Tagen die Eskalation ausgeführt werden soll. Im Standard wird eine Eskalation ausgeführt, wenn eine Mappe 2 Tage in einer Aktion liegt. Die hier angegebene Anzahl an Tagen wird auf das Eingangsdatum in die Aktion ("ActionStart") aufaddiert.
3. Bei der Addition der Tage wird im Standard der Arbeitskalender berücksichtigt. Andernfalls würden Rechnungen die an einem Freitag in die Aktion laufen direkt nach dem Wochenende eskalieren.

```
57          case "Action":
58              var escParam   = df.getParamObject("Escalation_WorkflowAction");
59              var doEscalate = df.getParamValue(escParam, 1, true);
60              if( doEscalate === false ){
61                  this.Log.info("["+id+"] checkEscalation() .. do not escalate workflow action .. return false");
62                  return false;
63              }
64              else{
65                  var noDays        = df.getParamValue(escParam, 2, 2);
66                  var useWorkCal    = df.getParamValue(escParam, 3, true);
67                  var actionStartTS = df.ActionStart;
68                  if( (actionStartTS instanceof Date)===true ){
69                      var startStr = util.convertDateToString(actionStartTS, "dd.mm.yyyy HH:MM:SS");
70                      var escDate  = context.addTimeInterval(actionStartTS, Number(noDays), 'days', useWorkCal);
71                      if( escDate < new Date() ){
72                          this.Log.info("["+id+"] checkEscalation() .. action started("+startStr+") .. return true");
73                          return true;
74                      }
75                      else{
76                          this.Log.info("["+id+"] checkEscalation() .. action started("+startStr+") .. return false");
77                          return false;
78                      }
79                  }
80              }
81              break;
82      }
```

Die Eskalation erfolgt in der Standardauslieferung immer an die aktuell sperrenden Benutzer. Für eine Eskalation zum Beispiel an Vorgesetzte müsste ein neuer Eskalations-Typ definiert werden.

Die Eskalation kann über das Skript "Invoice_Job_EscalateActionStart" gesteuert werden.

Eskalation zur Zahlungsbedingung ("PPC")

Über den Job "Invoice_JOB_EscalatePPC" werden die Zahlungsbedingungen eskaliert. Das Skript iteriert alle Rechnungs-Mappen und ermittelt zunächst für alle Rechnungen das Datumswerte zur hinterlegten Zahlungsbedingung. Im Anschluss werden die Datumswerte "ConditionsOfPayment_Net" und "ConditionsOfPayment_Date1" geprüft. Auf die Datumswerte werden in der Standard-Auslieferung 2 Tage ohne Berücksichtigung des Arbeitskalenders addiert. Sollte eines der berechneten Datumswerte nicht mehr in der Vergangenheit liegen muss eskaliert werden.

```
84          case "PPC":
85              var checkDate    = context.addTimeInterval(new Date(), 4, 'hours', false);
86              var checkDateStr = util.convertDateToString(checkDate, "dd.mm.yyyy HH:MM");
87              df.setPaymentConditions(true);
88              var escDays = 2;
89              var netDate = df.ConditionsOfPayment_NetDate;
90              if( netDate instanceof Date ){
91                  var netDateStr = util.convertDateToString(netDate, "dd.mm.yyyy");
92                  if( netDate <= checkDate ){
93                      var escDate = context.addTimeInterval(netDate, escDays, 'days', false);
94                      if( escDate >= checkDate ){
95                          this.Log.info("["+id+"] checkEscalation() .. netDate("+netDateStr+") + "+escDays+"days >= checkDate("+checkDateStr+")");
96                          return true;
97                      }
98                      else{
99                          this.Log.info("["+id+"] checkEscalation() .. netDate("+netDateStr+") + "+escDays+"days < checkDate("+checkDateStr+")");
100                     }
101                 }
102                 else{
103                     this.Log.info("["+id+"] checkEscalation() .. netDate("+netDateStr+") > checkDate("+checkDateStr+")");
104                 }
105             }
106             else{
107                 /* this.Log.info("["+id+"] checkEscalation() .. df.ConditionsOfPayment_NetDate is not an instance of Date"); */
108             }
109             var percDate = df.ConditionsOfPayment_Date1;
110             if( percDate instanceof Date ){
111                 var percDateStr = util.convertDateToString(percDate, "dd.mm.yyyy");
112                 if( percDate <= checkDate ){
113                     var escDate = context.addTimeInterval(percDate, escDays, 'days', false);
114                     if( escDate >= checkDate ){
115                         this.Log.info("["+id+"] checkEscalation() .. percDate("+percDateStr+") + "+escDays+"days >= checkDate("+checkDateStr+")");
116                         return true;
```

Die Eskalation erfolgt an den bzw. die sperrenden Benutzer.