Workflows und Workflow Management Systeme

Wolfgang Hüttig

26. Juni 2009

Zusammenfassung

Workflows und Workflow Management Systeme spielen eine immer größere Rolle in Business Prozessen aber auch im Wissenschaftlichen Bereich. Hier wird euch eine kurze Einführung in Business Workflows und Workflow Management Systeme für Business Prozesse gegeben und auch die Unterschiede zu Scientific Workflows kurz angesprochen. Des weiteren enthaltet ihr einen Einblick in die verschiedenen Komponenten eines Workflow Management Systems und wie diese zusammen arbeiten.

Inhaltsverzeichnis

1 Einleitung

Dieser Abschnitt gibt eine kurze Einleitung darüber, wofür Workflows verwendet werden und darüber hinaus, wie sie in SIMPL Verwendung finden werden.

1.1 Wofür Workflows?

Workflows (zu Deutsch: Arbeitsabläufe) können als eine Sequenz von Arbeitsschritten beschrieben werden. Diese Schritte können von einer Person, von Personengruppen oder auch von Maschinen ausgeführt werden. Die Ausführung erfolgt dabei sequenziell, wobei verschiedene unabhängige Arbeitsschritte jedoch parallelisiert werden können.

Im Business-Bereich werden Workflows überall dort eingesetzt wo sich Prozesse leicht automatisieren lassen und die Prozesse häufig ausgeführt werden müssen. Das spart Zeit, Arbeitskräfte und somit Geld. Business Workflows werden zum Beispiel in Banken eingesetzt wenn es um die Kreditvergabe geht. Somit kann über einen Kredit unter einem bestimmten Betrag (z.B. < 5000 Euro) in Millisekunden entschieden werden, ob dieser gewährt wird oder nicht.

Im Scientific-Bereich werden Workflows zum Beispiel bei Simulationen eingesetzt. Hier müssen Unmengen von Daten aus verschiedensten Datenquellen verarbeitet werden. Oft ändern sich dabei nur die Daten und nicht das zugrunde liegende Modell und somit kann eine Simulation mehrfach hintereinander mit verschiedenen Daten ausgeführt werden.

1.2 Workflows in SIMPL

Im Projekt *SIMPL* geht es darum den Zugriff auf verschiedene Datenquellen (z.B. Sensordaten, Datenbanken, usw...) über BPEL(Business Process Execution Language) zu ermöglichen. BPEL ist eine Sprache zum modellieren von Business-Workflows, doch kamen im laufe der Zeit auch Funktionalitäten hinzu, welche es ermöglichen Scientific-Workflows zu modellieren. Der Zugriff auf verschiedene Datenquellen ist hier jedoch ausschließlich über Web Services möglich. Im Rahmen von SIMPL, soll die Sprache BPEL analysiert werden und um diese Funktionalitäten erweitert werden, die benötigt werden um statisch, während der Modellierung, aber auch dynamisch, während des Deployments der Software beliebige Datenquellen in den BPEL-Prozess einbinden zu können.

2 Workflows

> *Workflow is concerned with the automation of procedures where documents, information or tasks are passed between participants according to a defined set of rules to achieve, or contribute to, an overall business goal.* - D. Hollingsworth [1]

2.1 Was sind Workflows?

Workflows und Prozesse werden nicht selten als Synonyme gesehen. Wir unterscheiden hier jedoch zwischen diesen beiden Begriffen. Den Begriff Prozess,

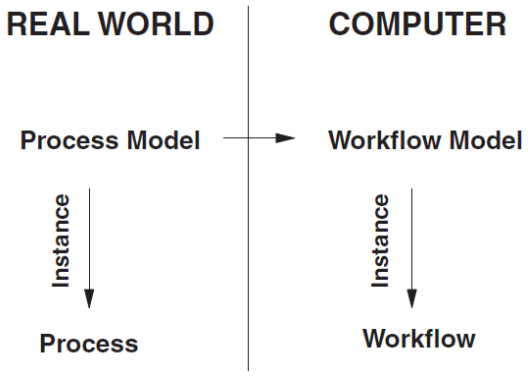

Abbildung 1: Reale Prozesse und Workflows [2]

beziehen wir auf real ablaufende Prozesse, welche durch Menschenhand und nicht durch den Computer oder Maschinen übernommen werden. Dem gegenüber stehen die Workflows, welches ebenfalls Prozesse sind, die jedoch von Computern automatisiert übernommen werden(siehe Abbildung 1). Somit können zum Beispiel, je nach Komplexität, Teile oder sogar ganze Prozesse von Workflows übernommen werden. Es gibt verschiedene Methoden Workflows zu modellieren. Häufig Verwendung finden hier zum Beispiel Petri Netze, State-Chart-Diagramme, Entity-Relationship-Diagramme und die gerichteten Graphen.

Workflows besitzen drei von einander unabhängigen Dimensionen. Deshalb werden sie oft als Würfel dargestellt (siehe Abbildung 2)[2].

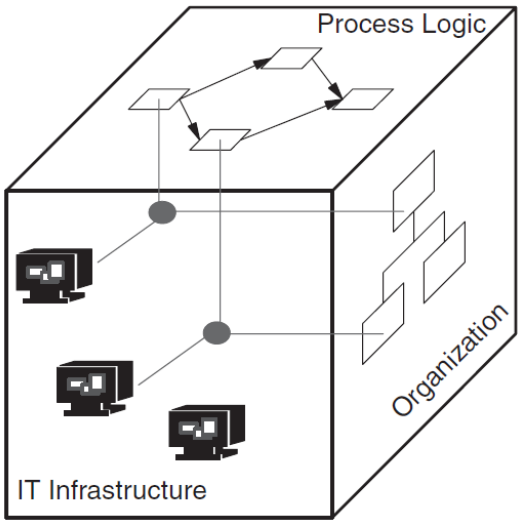

Abbildung 2: Dimensionen von Workflows [2]

Die erste Dimension beschreibt die Logik der Prozesse. Sie zeigt welche Aktivitäten vorhanden sind (jedes Kästchen repräsentiert ein Programm eine Aktivität oder einen weiteren Prozess) und in welcher Reihenfolge sie ausgeführt

werden. Die Pfeile geben dabei die Richtung an, in welche der Prozess läuft.

Die zweite Dimension des Würfels beschreibt die Struktur des Unternehmens. Hier werden, z.B. durch Organigramme, Abteilungen, Rollen und auch einzelne Mitarbeiter beschrieben. Mit diesen Angaben kann man nun bestimmten Abteilungen, Rollen oder Personen einen Prozess zuordnen.

Die dritte Dimension beschreibt die IT-Infrastruktur des Unternehmens auf dem das Workflow laufen soll. Sie beschreibt welche Ressourcen(Programme/Workflows) auf welchen Rechnern ausgeführt werden.

2.2 Arten von Workflows

Im Groben gibt es vier Arten von Workflows die man nach ihrem Geschäftsnutzen und der Anzahl der Wiederholungen unterscheiden kann. Das Schaubild (Abb. 3) soll dies verdeutlichen [2].

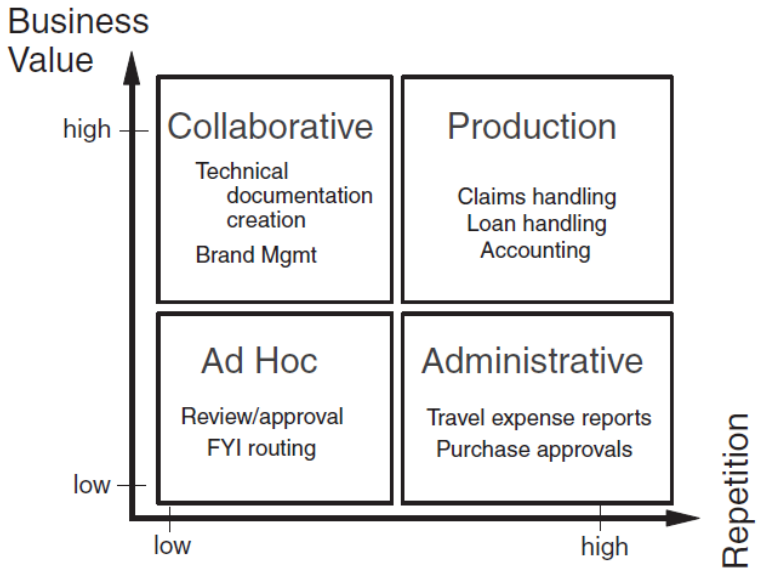

Abbildung 3: Arten von Workflows

Der Geschäftsnutzen gibt dabei an wie wichtig der Prozess für eine Firma ist. Ein Hoher Geschäftsnutzen heißt, das es sich dabei um das Kerngeschäft handelt. Zum Beispiel das herstellen von Dosen in einer Dosenfabrik.
Die Wiederholrate gibt an wie oft eine Aktion wiederholt werden muss. Außerdem gibt sie an wie sehr es sich lohnt für das entsprechende Problem ein Modell anzufertigen. Denn das Erstellen eines Modells ist eine sehr arbeitsaufwändige und kostenintensive Tätigkeit. Im Folgenden die vier Unterscheidungen [2].

- *Ad hoc-Workflows*: *Ad hoc-Workflows* haben einen geringen Geschäftsnutzen und eine geringe Wiederholrate. In ihrer Ausführung haben sie viele

Freiheitsgrade, da jeder Arbeitsschritt von allen beteiligten Personen festgelegt wird.
Eingesetzt werden Ad hoc-Workflows um beispielsweise spontane Arbeitsprozesse mit verschiedenen Teilnehmern zu koordinieren.

- *Collaborative-Workflows*: *Collaborative-Workflows* haben einen hohen Geschäftsnutzen, werden jedoch nur wenige male ausgeführt. Ein Beispiel hierfür ist zum Beispiel die Erstellung einer Dokumentation für ein Software-Projekt.

- *Administrative-Workflows*: *Administrative-Workflows* haben einen geringen Geschäftsnutzen, aber eine hohe Anzahl an Wiederholungen. Ein Beispiel für diese Art von Workflows sind Spesenabrechnungen. Der Mitarbeiter füllt ein Formular aus, gibt dieses an seinen Vorgesetzten, der es bestätigt. Nachdem die Buchhaltung die Richtigkeit überprüft hat, tätigt sie die Überweisung.

- *Production-Workflows*: *Produktion-Workflows* haben einen hohen Geschäftsnutzen und eine hohe Wiederholrate. Sie sind die Workflows die das Kerngeschäft eines Unternehmens darstellen. Ein Beispiel hierfür wäre die Kreditvergabe in einer Bank, wo für niedrige Beträge (z.B $<$5000 Euro)in Sekundenbruchteilen über eine Kreditvergabe entschieden werden kann, da dies an fest vorgeschriebene Muster gebunden ist.

2.3 Business Workflows vs Scientific Workflows

In der Basis sind sich Business-Workflows und Scientific-Workflows ähnlich. Beide beschreiben Sequenzen von wiederholten Aktivitäten und die Aufgaben die damit verbunden sind. Darüber hinaus gibt es in beiden Workflow-Typen Aktivitäten, die nicht von Maschinen ausgeführt werden können, sondern noch immer auf die Mithilfe von Menschen angewiesen sind, sich also nicht vollständig automatisieren lassen. Viel interessanter ist jedoch worin sich die beiden Workflow-Typen unterscheiden.

Zum einen wäre da die Art der *zu verarbeitenden Prozesse* zu sehen. In *Business-Workflows* werden hauptsächlich Informationen verarbeitet und damit definierte Aufgaben ausgeführt und dabei immer im Blick die Richtlinien und Regeln des Unternehmens mit dem Ziel den Gewinn des Unternehmens zu steigern. Auf der *Scientific-Workflow* Seite, werden größtenteils Informationen und Daten analysiert, es werden Experimente durchgeführt, und Daten werden auf Grund der Ergebnisse verändert. Das Ziel der Workflows ist, anders als bei den Business-Workflows, natürlich das Lösen von Problemstellungen.

Im *Business-Workflow* Bereich haben wir größtenteils statische Prozeduren, welche meist über einen langen Zeitraum ihre Gültigkeit behalten, so lange sich am darunterliegenden Prozess nichts ändert. Das kommt vor, wenn sich zum Beispiel Richtlinien innerhalb der Firma ändern oder die Firma durch äußere Einflüsse dazu gezwungen wird. Darüber hinaus sind alle Änderungen an den Prozeduren durch das Management abzusegnen. Im *Scientific-Workflow* Bereich

ist die Sache genau andersherum. Hier wird größtenteils mit dynamischen Prozeduren gearbeitet, da im wissenschaftlichen Bereich nicht selten auch mit einem trial and error Ansatz gearbeitet wird, in dem Prozeduren so lange durch ausprobieren verändert werden, bis sie die gewünschten Ergebnisse liefern. Hinzu kommt das Wissenschaftler die Prozeduren in Eigenverantwortung ändern und die Änderungen nicht gegenüber eines Managers rechtfertigen müssen.

Die Daten und vor allem die Datenmengen unterscheiden sich in beiden Typen erheblich. Im *Business-Workflow* Bereich, haben wir es oft mit Formular Daten zu tun, die von einem Nutzer irgendwo eingegeben und dann mit einer Datenbank abgeglichen und in diese Geschrieben werden. Die verarbeitete Datenmenge ist hierbei in den meisten Fällen überschaubar. Im *Scientific-Workflow* Bereich wird mit äußerst heterogenen Daten gearbeitet, wobei vorher oft noch gar nicht klar ist was und vor allem wieviele Daten verarbeitet werden müssen. Im Bereich von Beispielsweise Simulationen müssen Unmengen von Daten ausgewertet werden die aus den Verschiedensten Bereichen kommen können.

Zuletzt geht es noch um die Ersteller der Workflows. *Business Workflows* werden meist von professionellen Software- und Business-Flow Experten [3] erstellt, sie sich mit der Materie und den unterliegenden Sprachen und Werkzeugen besten auskennen. *Scientific-Workflows* werden oft vom Wissenschaftler selbst erstellt, und von der Erstellung von Workflows oft nicht viel Ahnung haben. Deshalb benötigen sie zum Beispiel besondere Benutzer Interfaces, welche es ihnen erlauben schnell und vor allem einfach Workflows zu erstellen und auszuführen.

3 Workflow Management Systeme

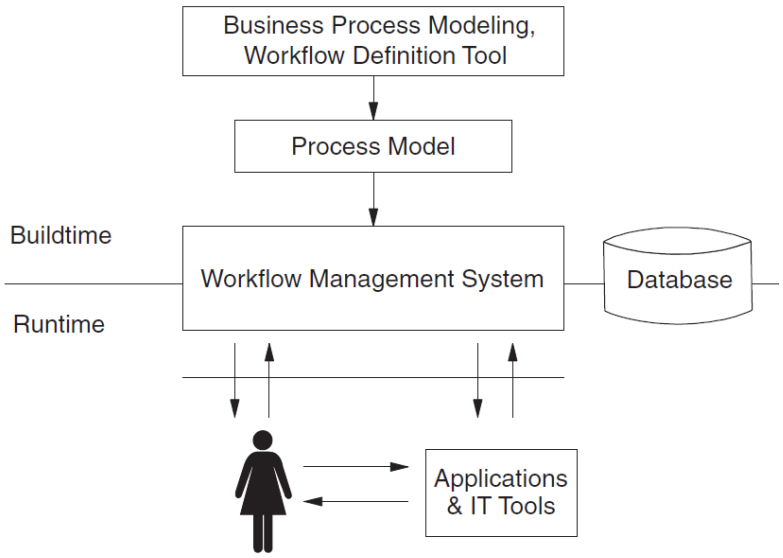

Abbildung 4: Struktur eines WfMS [2]

Workflow Management Systeme(WfMS) sind prozessorientierte Systeme, die

Prozesse(Workflows) nach einem vorher definierten Modell ausführen. In Workflow Management Systemen unterscheiden wir zwischen den vier Hauptkomponenten.[2]

- *Metamodell*: Definiert die Strukturen die von dem WfMS unterstützt werden, zum Beispiel die Struktur der Prozessmodelle.
- *Buildtime*: Hier wird der Ablauf oder besser das Workflow Definiert
- *Runtime*: Das Workflow wird ausgeführt und die Ausführung zusätzlich überwacht.
- *Datenbank*: Die Datenbank enthält alle wichtigen Informationen die von den Buildtime- und Runtime-Komponenten gebraucht wird so zum Beispiel die Nutzer daten.

3.1 Buildtime

Die Buildtime-Komponente bietet alle Funktionalität um die Informationen für das Workflow zu erstellen, zu Testen und zu verwalten. Zusätzlich hat sie Verbindung zur Datenbank um alle erhaltenen Informationen dort abzuspeichern. Im Groben gibt es zwei Möglichkeiten wie die Buildtime-Komponente aussieht. Eine Möglichkeit ist die Informationen über ein Grafisches User Interface zu definieren. Die andere Möglichkeit ist, dass die Informationen über eine Flow-Definition-Language(FDL) eingegeben werden. Beide Möglichkeiten werden im Folgenden vorgestellt.

Graphical User Interfaces(GUI): In einer GUI werden Aktivitäten als Symbole dargestellt und können ganz einfach auf einem virtuellen Arbeitspapier platziert und bearbeitet werden. Danach werden alle platzierten Aktivitäten noch mit Pfeilen verbunden, welche die Reihenfolge angibt, in der das Workflow abgearbeitet wird. Abbildung 5 zeigt einen Kreditvergabe Prozess in der Buildtime Komponente von MQSeries Workflow

Flow Definition Language: Bei der Flow-Definition-Language(FDL) handelt sich um eine Skriptsprache die in vielen Fällen zum Beispiel auf XML aufbaut. Dies ermöglicht eine einfache Definition von verschiedenen Workflow-Teilen, wie zum Beispiel dem Prozess Modell und der Unternehmensstruktur. Auch lassen sich diese schnell mit einander Austauschen. Der Vorteil einer FDL gegenüber eines GUI ist, das sich nur hier auch komplexeste Workflows definieren lassen, da eine GUI nicht dazu in der Lage ist die selbe Mächtigkeit in der Beschreibung eines Workflows zu erreichen, wie eine FDL. Anders als eine GUI repräsentiert eine FDL das dem Workflow zugrunde liegende Metamodell. Im Folgenden ein kleiner Teil der Kreditvergabe aus Abbildung 5 als Code der MQSeries Flow-Definition-Language. (Beispiel entnommen aus [2])

```
PROCESS 'Loan Process'

    PROGRAM_ACTIVITY 'Collect Credit Information'
        PROGRAM 'Collect Credit Information Program'
```

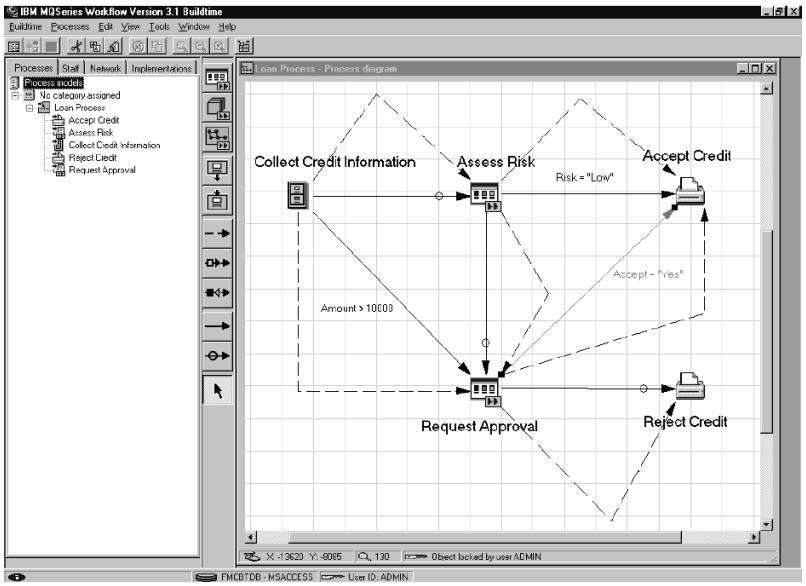

Abbildung 5: Grafisch Buildtime Komponente.(MQSeries Workflow)[2]

```
        DONE_BY 'Loan Officer'
    END 'Collect Credit Information'


    PROGRAM_ACTIVITY 'Assess Risk'
        PROGRAM 'Assess Risk Program'
        DONE_BY 'Financial Officer'
    END 'Assess Risk'

    CONTROL FROM 'Collect Credit Information'
        TO 'Assess Risk'
END 'Loan Process'

PROGRAM 'Collect Credit Information Program'
    WINNT
        EXE
            PATH_AND_FILENAME 'CCI.EXE'
END 'Collect Credit Information Program'

PROGRAM 'Assess Risk Program'
    AIX
        EXE
            PATH_AND_FILENAME 'AR.EXE'
END 'Assess Risk Program'

ROLE 'Financial Officer'
```

```
    RELATED_PERSON MMayer
END 'Financial Officer'

ROLE 'Loan Officer'
    RELATED_PERSON MRoss
END 'Loan Officer

PERSON MMayer
    FIRST_NAME Mike
    LAST_NAME Mayer
END MMayer

PERSON MRoss
    FIRST_NAME Mary
    LAST_NAME Ross
END MRoss
```

Hier schön zu sehen ist, dass der Programmcode alle Informationen des Modells enthält(Prozess-Struktur und Organisations-Struktur. Zu allererst werden die Aktivitäten Definiert, von welchen Programmen sie übernommen werden und wer die Ausführende Rolle ist. direkt darunter werden die Programme genau definiert mit ihrem Pfad und der auszuführenden Datei. Zu guter letzt folgen noch die Zuordnung von Bestimmten Personen zu einer Rolle und dann zum Schluss noch die Beschreibung der Personen, in diesem Fall durch vor und Nachnamen.

3.2 Metamodell

Das Metamodell deckt die 3 Dimensionen der Workflow ab. Es ist dazu da, die Prozessmodelle, Organisationsstrukturen und Infrastruktur des Workflow-Management-Systems zu beschreiben.

3.2.1 Organisationsmodell

Es gibt mehrere Möglichkeiten Aktivitäten in Workflows zu kategorisieren. Eine Möglichkeit wäre zum Beispiel die Art des Ausführens. Dabei gibt es Grundlegend erst einmal zwei Kategorien. Zum einen sind es manuell ausgeführte Aktivitäten. Sie werden einer bestimmten Person zugesprochen und können auch nur auf die Anfrage der Person ausgeführt werden.

Anders sind da wiederum die automatisch ausgeführten Aktivitäten. Sie werden vom System selbst übernommen und ausgeführt. Dabei ist zu beachten, das automatische Aktivitäten trotzdem einer Person oder einer Rolle zugeordnet sind und das System die Aktivitäten in deren Namen ausführt. Das hat zur Folge das der Name der entsprechenden Person im Business Prozess und im Workflow Management System bekannt sein muss.

Die Aktivitäten einer festen Person zu zuordnen hat jedoch einen großen Nachteil. Verlässt die Person die Firma oder wird einfach durch eine andere

ersetzt, muss der Prozess jedes mal neu angepasst werden. Besser ist es die Aktivität einer Abteilung, einer Rolle oder einer Position zuzuordnen. So kann man zum Beispiel eine Aktivität dem Abteilungsleiter zuordnen. Wenn die Aktivität ausgeführt wird, sucht sich das Workflow-Management-System selbst heraus, wer gerade Abteilungsleiter ist und weißt diesem die Arbeit zu. Sollte der Abteilungsleiter zwischenzeitlich wechseln, wird die Aktivität automatisch der andere Person zugewiesen. Dies passiert über Auflösung der Mitarbeiter, auf Namen der Personen, weshalb es dir Organisationsdaten in dieser Form gibt.

Es reicht aber nicht, dass das WfMS die Organisationsstrukturen kennt, sondern es muss auch Dinge wissen wie: Wer ist der Vorgesetzte von wem. Denn wenn eine Person eine Aktivität ausführen soll und der nächste Schritt soll von dessen Vorgesetzten ausgeführt werden, so müssen diese Informationen ebenfalls im WfMS bekannt sein.

In den verschiedenen WfMS gibt es verschiedene Ansätze von Organisationsmodellen. Zum einen gibt es das *statische Organisationsmodell*. In diesem sind die Einträge und ihre Beziehung zu einander unveränderbar.

Dem gegenüber steht das *dynamische Organisationsmodell*. Das zu Grunde liegende Organisationsmodell(Metamodell) kann hier verändert werden oder durch ein neues ersetzt werden. (Metamodelle können z.B. Entity-Relationship Diagramme sein)

Zur Ablage und Verwaltung der Organisationsdaten, wenn mehrere WfMS auf die selben Daten zugreifen wollen, gibt es drei Ansätze: [2]

1. Jedes WfMS hat seine eigene Datenbank in dem die Organisationsdaten gespeichert werden. Der Vorteil daran ist, die Datenbank kann für das entsprechende WfMS optimiert werden und es hat keinen Einfluss auf die anderen beteiligten Systeme. Der Nachteil ist, dass wenn man etwas an den Organisationdaten ändert, dies auch in den Datenbanken aller anderen beteiligten WfMS ändern muss und es somit zu Inkonsistenzen zwischen den verschiedenen Organisationsdaten kommen kann.

2. Alle beteiligten WfMS teilen sich ein gemeinsames Verzeichnis, in dem die Organisationsdaten abgelegt werden und jedes der WfMS kann diese Daten bearbeiten. Das hat den Vorteil, dass man nur in dieser einen Datenbank etwas ändern muss und es somit keine Konsistenzprobleme gibt.

3. Auch hier gibt es ein Gemeinsames Verzeichnis, auf das die WfMS jedoch nur zugreifen können. Die Daten werden von einem anderen System verwaltet. Vorteil ist auch hier, dass nur ein Speicherort für mehrere WfMS vorhanden sein muss. Nachteile beim zweiten und dritten Ansatz sind jedoch, dass das gemeinsame Verzeichnis einen Performance-Engpass darstellt. Dazu kommt, dass nur in den seltensten Fällen, alle WfMS das gleiche Organisationsmodell benutzen, somit muss gewährleistet werden, das jedes WfMS die Organisationsdaten auf sein eigenes Organisationsmodell übertragen kann.

Dies hat zur Folge das Anfragen gegen die Datenbanken verschieden Strukturiert sein müssen. Verwaltet das WfMS seine Daten selbst, so kann die Anfrage direkt

ausgeführt werden. Soll die Anfrage jedoch gegen ein anderes System ausgeführt werden, so muss die Anfrage in der *query language* des verwaltenden Systems gestellt werden.

3.2.2 Prozessmodell

Das Prozessmodell im allgemeinen beschreibt die Struktur eines Business-Prozesses in der Realität. Es beschreibt alle Pfade durch den Prozess und Regeln welche definieren wann welcher Pfad genommen werden soll. Dieses Modell ist eine Blaupause für den Prozess und wird benutzt um diesen zu erstellen. Hierbei ist es noch einmal wichtig zu verstehen, dass Prozesse nicht auf einem Computer ablaufen müssen. Es gibt auch heutzutage noch genug Business-Prozesse, die komplett ohne die mithilfe von Computern ausgeführt werden. Ein Beispiel hierfür wären Rundschreiben innerhalb einer Firma, die immer noch nicht überall über Mails abgewickelt werden. Ein Mitarbeiter bekommt das Schreiben, liest es, unterschreibt und reicht das Schreiben an jemanden weiter der noch nicht darauf unterschrieben hat.

Im nachfolgenden Abschnitt beschäftigen wir uns dann mit einer Art von Prozessmodellen, den Prozessgraphen.

3.2.3 Prozessgraph

Die heute gebräuchliche Art der Prozessmodelle sind *Prozess Graphen.* Der Hauptgrund dafür ist, dass sie einfach zu erstellen und zu verstehen sind und trotzdem eine große Aussagekraft besitzen(Abbildung 5 auf Seite 9 zeigt einen solchen Graphen). Hierbei handelt es sich immer um einen gerichteten azyklischen Graphen. Das Konzept des Prozessgraphen findet zum Beispiel auch in BPEL Anwendung.

Prozessmodelle sind hauptsächlich eine Sammlung von Aktivitäten, Kontrollverbindungen, Übergangsbedingungen, Input-Containern, Output-Containern und Datenverbindungen zusammen. Alle Komponenten werden im folgenden näher erläutert.

Container: Prozesse und Aktivitäten bekommen, wenn sie ausgeführt werden, Daten mitgegeben und diese werden in den Input-Containern gespeichert. Genauso kann jeder Prozess und jede Aktivität Daten zurück geben, welche dann im Output Container gespeichert werden. Als Beispiel wäre hier wieder die Kreditvergabe zu nennen. Wenn der Prozess gestartet wird, bekommt er beispielsweise die Kundennummer des Kunden übertragen, welcher einen Kredit beantragen möchte. Der Prozess übergibt die Kundennummer an eine Aktivität, in dem Fall ein Programm, welches die Datenbank anhand der Nummer nach allen relevanten Informationen des Kunden durchsucht und zurückgibt. Diese Daten werden dann Output-Container der Aktivität gespeichert. Der Inhalt des Output Containers kann später von anderen Aktivitäten benutzt werden oder als Entscheidung dienen, welcher weg im Graph weiter genommen wird.

Jedem Container wird dabei eine bestimmte Datenstruktur zugeordnet. Das kann ein String sein, ein Float oder auch eine Sammlung verschiedener Daten, also ein Array.

Aktivitäten werden im Schaubild als Kreise mit einem Namen dargestellt. Der Name der Aktivität gibt dabei meist Aufschluss über ihre Tätigkeit. Wir unterscheiden mehrere Arten.

- *Programm-Aktivität*: Programm-Aktivitäten werden von ihnen zugewiesenen Programmen ausgeführt.
- *Prozess-Aktivitäten*: Prozess-Aktivitäten stoßen Ihrerseits einen neuen Prozess an (Subprozess). Der Prozess pausiert so lange bis sein Subprozess vollständig beendet ist.
- *Block-Aktivität*: Die Block Aktivität ist ein Makro(Prozess), welches so oft ausgeführt wird bis eine end Bedingung erreicht ist.

Auf die Aktivitäten wird später noch einmal etwas detaillierter eingegangen.

Die *Kontrollverbindungen* werden im Schaubild als durchgezogene Linien gezeigt. Wenn dabei Verbindungen die Aktivität verlassen so nennen wir dies eine *Fork-Aktivität*. Oft ist dies auch ein Punkt ab dem parallel gearbeitet wird. Aktivitäten die von mehreren Verbindungen das Ziel sind, nennen wir *Join-Aktivität*. Eine Aktivität ohne eingehende Verbindungen wir Startaktivität genannt und eine Aktivität ohne ausgehende Verbindungen nennen wir Endaktivität.

Jede Kontrollverbindung hat darüber hinaus ein boolean Prädikat (Übergangsbedingung), welches entscheidet ob die Kante weiter verfolgt wird. Ein solches Prädikat wäre beispielsweise, ob ein Kredit von über 5000 Euro beantragt wurde. Wenn ja wird das Prädikat auf true gesetzt und der Pfad weiterverfolgt. Ansonsten wird das Prädikat auf false gesetzt. In diesem Falle muss geschaut werden, welche Art Aktivität die Verbindung als Ziel hatte. Handelt es sich dabei nämlich nicht um eine *Join-Aktivität*, werden von dem Ziel ebenfalls alle ausgehenden Verbindungen auf false gesetzt. Ist das Ziel der Verbindung eine *Join-Aktivität*, so wird an dieser gewartet bis alle eingehenden Verbindungen einen wert haben (true oder false). Haben alle eingehenden Verbindungen einen Wert, so wird an der Aktivität die *Join-Bedingung* (auch: Startbedingung) geprüft. Im einfachsten Fall könnte diese sein, dass alle eingehenden Verbindungen oder mindestens eine, true zeigen müssen.

Ist die *Join-Bedingung* true, wird die Aktivität ganz normal ausgeführt. Ist sie jedoch false, so wird die Aktivität übersprungen und alle ausgehenden Verbindungen der Ziel Aktivität werden auch auf false gesetzt. Dieses Vorgehen wiederholt sich so lange bis ein Endaktivität erreicht wurde oder es zum Beispiel an einer weitern Fork-Aktivität ankommt und dort anhält. Dieses vorgehen nennt sich "death path elimination"[2]. Dadurch wird sichergestellt, das der Prozess immer terminiert. Ein prozess wird dann beendet, wenn alle Endaktivitäten besucht worden sind (ausgeführt oder übersprungen).

Datenverbindungen werden im Schaubild als gestrichelte Linien dargestellt. Sie werden benötigt weil Input- und Output-Container nur lokale Daten enthalten. In dieser Verbindung ist spezifiziert, welche Daten(Felder) des Outputcontainer in den Input-Container der nächsten Aktivität kopiert werden. Dabei

reicht es aber nicht zu Spezifizieren, welche Daten kopiert werden sollen, denn der Aufbau des Input-containers der Zielaktivität ist meist ein anderer als der Aufbau der Start-Aktivität. Somit muss entschieden werden welches Feld des Output-Containers auf welches Feld des Input-Containers gemappt wird und welche Daten dabei umgewandelt werden müssen. Datenverbindungen müssen nicht zwingend zwischen Output- und Input-Container zwei verschiedener Aktivitäten bestehen, sie können auch zwischen Input- und Output-Container der selben Aktivität definiert werden. Das hat den Vorteil das nicht immer alle, vielleicht nicht benötigten Daten, vom Input- in den Output-Container geschrieben werden müssen. Darüber hinaus gibt es die Möglichkeit für jedes Feld in den Containern, Standardwerte zu definieren.

Aktivitäten im Detail bestehen noch einmal aus verschiedenen Bestandteilen die wir uns hier jetzt kurz anschauen wollen.

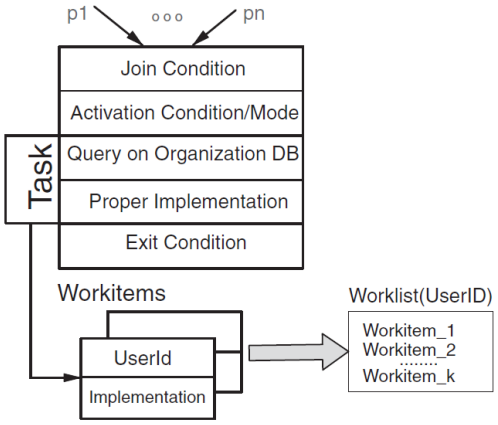

Abbildung 6: Detailliere Sicht auf Aktivitäten [2]

Die *Join-Bedingung* wurde bereits weiter oben beschrieben so gehen wir hier nicht weiter darauf ein.

Als nächstes wird die *Aktivierungsbedingung* geprüft. Diese gibt an, wann genau die Aktivität schließlich gestartet werden soll. Dies kann zum Beispiel bedeuten, das die Aktivität nicht Nachts gestartet wird, wenn die Mitwirkung einer Person erforderlich ist. In dem Fall würde die Aktivität eben erst nach Arbeitsbeginn starten. Der *Aktivierungsmodus* gibt an ob die Aktivität manuell oder automatisch gestartet werden soll. Manuel bedeutet das eine Person den Start der Aktivität anstoßen muss, zum Beispiel durch klicken auf ein entsprechendes Icon in der Arbeitsliste (siehe unten). Automatisch bedeutet, die Aktivität startet sofort. Dieser Modus findet dann Verwendung wenn keine Interaktion mit einer Person nötig ist.

Die beiden nächsten Punkte (Anfrage an Organisations-Datenbank und Implementierung), bilden zusammen eine *Task* und den Kern der Aktivität.

Bei der *Anfrage an die Organisations-Datenbank* wird geschaut, welche Personen einer Aktivität zugeordnet sind. (Mitarbeiter Auflösung). Oftmals sind mehrere Personen zum Ausführen der Aktivität angegeben. Das erhöht die Wahrscheinlichkeit das die Aktivität schnell ausgeführt wird. Das WfMS sorgt dafür, das nur eine der benachrichtigten Personen die Aktivität ausführen kann. Für jede dieser Personen wird ein Arbeitspaket(Workitem) erstellt, welches die Identifikation des Benutzers und die *Implementierung* der Aktivität enthält. Die Runtime-Umgebung des WfMS stellt eine Möglichkeit zur Verfügung, ähnliche Arbeitspakete in einer *Arbeitsliste (Worklist)* zu Organisieren. Je nachdem kann ein Arbeitspaket auch in mehreren Arbeitslisten vorkommen, zum Beispiel als Kreditantrag und als Aktivität mit hoher Priorität.

Die Implementierung kann ein Programm sein das gestartet wird oder ein weiterer Prozess oder Block der angestoßen wird. Ein Spezialfall ist es, wenn keine Implementierung angegeben ist, so nennt sich diese Form der Aktivität eine *Informationsaktivität* da diese zum Beispiel bedeutet, das ein Mitarbeiter einen Kunden anruft um Informationen von ihm zu erfragen.

Die *Endbedingung* entscheidet ob die Aktivität als Beendet gilt oder nicht. Ist die Endbedingung false, so wird die Aktivität noch einmal von vorne gestartet und nur bei true, wird die Navigation durch die Graphen fortgeführt. Ein Beispiel für eine Aktivität die eine solche Endbedingung nötig macht, wäre das Schreiben eines Briefes. Der Mitarbeiter fängt an einen Brief zu schreiben wird jedoch dazu gezwungen aufzuhören. Somit wird die Aktivität als nicht beendet markiert und wandert wieder in die Arbeitsliste. Von dort aus kann sie zu einem späteren Zeitpunkt wieder aufgerufen werden.

Am Anfang wurde erwähnt das es sich bei dem Graphen um einen azyklischen gerichteten Graphen handelt. Manchmal ist es jedoch nötig, das eine Aktivität mehrfach hintereinander ausgeführt wird, zum Beispiel bei der Kompilierung eines Programms, in dem mehrere Programmteile hintereinander und getrennt von einander Kompiliert werden sollen. Zu diesem Zweck wollen wir uns an dieser Stelle die *Blockaktivitäten noch etwas genauer anschauen*:

Blockaktivitäten sind im Grunde nicht anderes als Prozesse, mit der Besonderheit das sie durch ein Loop-Until-Konstrukt umgeben sind. Nach jedem durchlauf des Prozesses wird die Abbruchbedingung geprüft. Wird diese nicht erfüllt, wird die Aktivität ein weiteres mal ausgeführt. Wird sie erfüllt, beendet sich der Prozess ganz normal.

3.3 Runtime

Die Hauptaufgabe der Runtime-Komponente ist die ordnungsgemäße Ausführung der Prozesse. Dabei navigiert die Runtime-Komponente durch die Prozesse und interagiert mit dem Benutzer und Anwendungen.

Aktivitäten können als automatisch oder als Manuell definiert werden. Sind sie auf automatisch eingestellt, so starten sie von alleine. Bei Einstellung auf manuell, muss der Benutzer die Implementierung der Aktivität explizit starten.

Dabei kann die Implementierung auf dem System den Benutzers laufen, auf dem Prozessor auf dem auch das WfMS läuft. Die Implementierung kann aber auch auf einem ganz andere Prozessor ausgeführt werden. Mischformen von beidem gibt es auch, meist bei Client-Server-Anwendungen. Die Aktivitäten können dabei unbeaufsichtigt, also ohne Benutzerinteraktion, laufen oder beaufsichtigt. Auch verschiedene Datenbank aufrufe können während des Laufens stattfinden, z.B. Speichern oder Update von Informationen. Es kann aber auch sein, dass das WfMS nur Informationen sammelt die dann von Aktivität zu Aktivität weiter gegeben werden.

3.3.1 Der Prozess Lebenszyklus

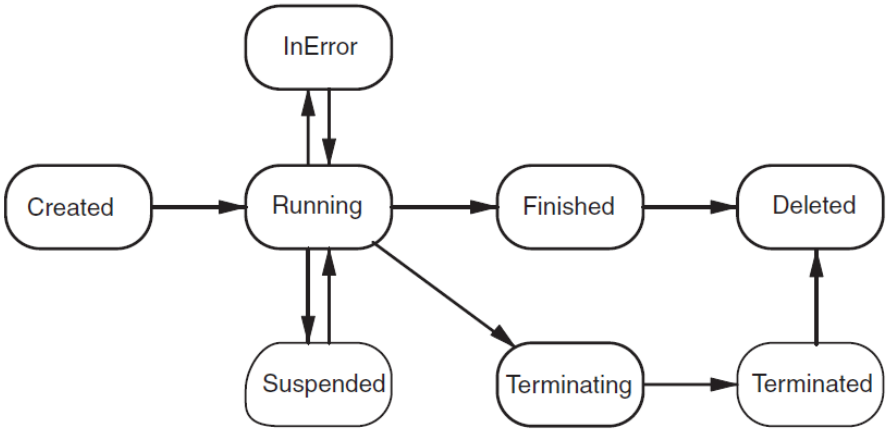

Abbildung 7: Der Prozess Lebenszyklus

Wenn ein Prozess erstellt wird, befindet er sich im *Erstellen*-Zustand. Wird er dann gestartet, entweder durch einen Benutzer oder automatisch als Subprozess, geht er in den *Läuft*-Zustand über. Dort bleibt er so lange er ausgeführt wird, wobei er durch das Prozessmodell Navigiert und Arbeitspakete erstellt werden. Dort bleibt der Prozess bis er beendet wird und in den *Beendet*-Zustand übergeht. Dort bleibt er bis er von einem Benutzer direkt gelöscht wird oder eine vom Benutzer festgesetzte Zeit vergangen ist. Danach geht er in den *Löschen*-Zustand über und wird gelöscht. Der Prozess kann von einem Benutzer jedoch auch zurückgestellt werden und geht in den *Aussetzten*-Zustand. Dort werden keine Aktivitäten mehr ausgeführt und keine Arbeitspakete erstellt, so lange bis der Prozess wieder aufgenommen wird. Wenn ein Prozess aus irgendeinem Grund nicht mehr gebraucht wird, kann er von befugten Benutzern gelöscht werden. Direkt geht der Prozess in den *Abbrechen*-Zustand über und nachdem alle Aktivitäten beendet worden sind, geht er schließlich in den *Abgebrochen*-Zustand und wird danach gelöscht. Für den Fall das während der Ausführung des Prozesses ein Fehler auftritt, gibt es noch einen *Fehler*-Zustand. In diesem Zustand ist der Prozess angehalten und Maßnahmen zur Fehlerbehebung können ergriffen werden.

3.3.2 Der Aktivitäten Lebenszyklus

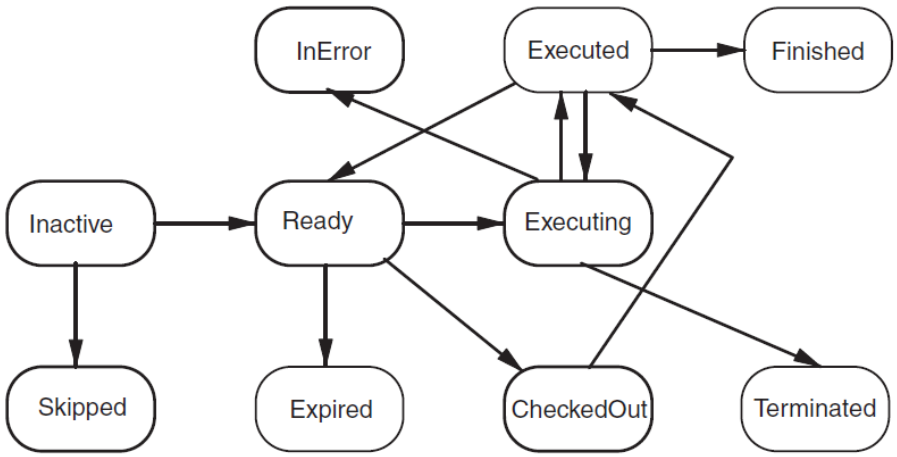

Abbildung 8: Der Aktivitäten Lebenszyklus

Sobald ein Prozess startet werden alle Aktivitäten innerhalb des Prozesses in auf den *Inaktiv*-Zustand gesetzt. In diesem Zustand werden sie auf entsprechende Anfragen hin angezeigt, zum Beispiel welcher Mitarbeiter welcher Aktivität zugeordnet wird. Wenn innerhalb der Aktivität die Mitarbeiterauflösung beendet ist und die passenden Arbeitspakete erstellt wurden, geht die Aktivität in den *Bereit*-Zustand über. Wenn ein Benutzer oder das system selbst die Implementierung der Aktivität aufruft, geht er in den *Ausführungs*-Zustand über, so lange bis der Subprozess oder das Programm beendet wurden. Danach erfolgt der Übergang in den *Ausgeführt*-Zustand. Ist für die Aktivität eine automatische Endbedingung definiert, geht sie sofort in den *Beendet*-Zustand über. War die Endbedingung auf Manuell eingestellt, wird auf die Bestätigung eines Benutzers gewartet. Danach geht die Aktivität, je nachdem was der Benutzer gewählt hat, in den Beendet- oder Bereit-Zustand über. Nachdem die Aktivität beendet wurde, folgt der Prozess allen ausgehenden Kontrollverbindungen der Aktivität.

Wenn eine Aktivität übersprungen wurde, weil der Pfad nicht gewählt wurde oder die *Join-Condition* fehlschlägt, geht die Aktivität in den *Überspringen*-Zustand über. Manchmal verliehen Aktivitäten auch ihre Gültigkeit, während des Ausführens und dann gehen sie in den *Abgelaufen*-Zustand über.

Eine Besonderheit bei Aktivitäten ist, das ein berechtigter Benutzer, sie außerhalb des Kontrollflusses von Prozessen ausführen kann. Er macht einen *Checkout* der Aktivität und dann die Implementierung dann jederzeit starten. Wenn die Ausführung fertig ist, geht die Aktivität in den Ausgeführt-Zustand über, so als wäre sie ganz normal innerhalb eines Prozesses ausgeführt worden.

Wenn das Programm welches ausgeführt werden soll nicht gefunden wird oder nicht richtig beendet wird, geht die Aktivität in den *Fehler*-Zustand über. Auch hier können nun wieder Maßnahmen zur Fehlerbehebung ergriffen werden. Zu guter letzt können Aktivitäten noch Abgebrochen werden wodurch sie in

den *Abgebrochen-Zustand* gehen. Abgebrochen werden Aktivitäten zum Beispiel, wenn der darüberliegende Prozess abgebrochen wird.

3.3.3 Arbeiten mit Prozessen

Eine Person die dazu berechtigt ist, kann aus der Liste der Prozessmodelle eines Auswählen und daraus einen neuen Prozess erstellen. Außerdem kann ein Prozess auch direkt Ausgeführt werden, wobei er erst erstellt wird und dann Ausgeführt wird. Sobald ein Prozess erstellt ist, hat er einen Eintrag in der Prozessliste. Von dort aus kann er, so fern noch nicht geschehen, gestartet werden. Er kann für eine bestimmte oder unbestimmte Zeit ausgesetzt werden und wieder aufgenommen werden. Darüber hinaus kann man anfragen an den Prozess stellen um zum Beispiel den Namen und die Beschreibung des Prozesses zu erhalten. Zu guter letzt kann man Prozessdaten noch aktualisieren.

Natürlich muss zuerst eine Session auf dem WfMS gestartet werden. Dafür Startet man den Client und gibt seinen Benutzernamen und Passwort ein. Der Benutzer bleibt so lange eingeloggt bis er sich selber ausloggt oder das WfMS die Verbindung nach einer Zeit wegen Inaktivität kappt.

Die Interaktion mit dem WfMS passiert über eine Gui, die die Grundlegenden Funktionen zur Verfügung stellt. Die wichtigsten wären hier zu nennen mit Arbeitslisten, die die Arbeitspakete enthalten. Darüber hinaus gibt es noch Prozess- und Aktivitätenlisten, wobei letzteres nur sehr selten gebraucht wird. Das wichtigste Arbeitsmaterial ist jedoch die Arbeitsliste oder in vielen Fällen auch mehrere Arbeitslisten. Für den Aufbau der Listen gibt es drei verschiedene Ansätze.

- *Pull-Mode*: Neue Arbeitspakete, welche von dem WfMS erstellt werden, tauchen nicht automatisch in den Arbeitslisten auf, sondern müssen vom Benutzer jedes mal explizit angefordert werden. Das hat den Vorteil, dass der Benutzer volle Kontrolle über den Aufbau seiner listen hat. Nachteilig ist das vielleicht wichtige Arbeitspakete nicht sofort bemerkt werden.

- *Push-Mode*: Neue Arbeitspakete werden sofort nach dem Erstellen auf die entsprechenden Arbeitslisten geschrieben (gepushed). Vorteil ist, das wichtige Arbeitspakete sofort sichtbar werden. Nachteilig ist, das Arbeitslisten einer ständigen Veränderung unterworfen sind und dadurch unübersichtlich wirken können.

- *Grab-Mode*: Das WfMS liefert die Arbeitspakete dann wenn sie gebraucht werden. Das heißt, so bald ein Arbeitspaket aus einer Liste abgearbeitet wurde, wird sofort ein neues Bereitgestellt und die Implementierung automatisch gestartet, so das ein Manuelles starten gar nicht mehr notwendig ist. Das hat vor allem den Vorteil in Listen die einer ganzen Gruppe von Leuten zugänglich ist. Im Pull-Mode kommt es dabei sehr häufig vor, das Pakete, bereits von einem andren Benutzer bearbeitet werden, wenn sie in der Liste stehen. Im Push-Mode kommt dies immer noch vereinzelt

vor, jedoch nicht so häufig wie im Pull mode. Im Grab Mode kann dieses Phänomen gar nicht mehr auftreten.

Die Benutzer kann nun auch verschiedene Optionen entweder auf Arbeitspaketen oder Aktivitäten ausführen. Typische *Funktionen auf Arbeitspaketen* sind Löschen es Paketes, so lange es mindestens noch einen anderen Benutzer gibt, in dessen Arbeitsliste das Paket vorhanden ist. Dann ist es Möglich, zum Beispiel wenn das Paket aus irgendeinem Grund nicht ausgeführt werden kann, dieses an einen anderen Benutzer zu übergeben. Zu guter letzt können Arbeitspakete auch noch für eine Bestimmte Zeit oder auch unendlich lange, auf Eis gelegt werden. Für die angegebene Dauer werden sie in der Arbeitsliste nicht mehr angezeigt.

Aktivitäten können zum Beispiel gestartet werden, wenn sie auf den Manuellen Modus eingestellt sind. Sie können neu gestartet werden wobei sie wieder in den Bereit-Zustand übergehen oder mann kann direkt die Implementierung noch einmal neu starten. Wenn das Beenden der Aktivität ebenfalls auf Manuell gestellt ist, kann dies hier auch gemacht werden. Der Subprocess der Aktivität kann zurückgestellt und später wieder aufgenommen werden oder die Implementierung kann vollständig abgebrochen werden.

Sobald eine Aktion von einem Benutzer gestartet wurde, wird sie in allen Arbeitslisten von anderen Benutzern deaktiviert.

Direkt auf Aktivitäten, wenn sie in der Aktivitätenliste stehen, arbeitet man nur sehr selten. Trotzdem gibt es ein paar Funktionen die man darauf ausführen kann:

- Aktivitäten für bestimmte Nutzer erstellen
- Aktivitäten neu ansetzten um ein neues Set an Arbeitspaketen zu erstellen.
- Infos über die Aktivität abrufen, so zum Beispiel den Status.
- Container Reparieren
- Beenden erzwingen wenn ein Container Repariert wurde.

Literatur

[1] D. Hollingsworth *The Workflow Reference Model*, Workflow Management Coalition, 19.Jan.95.

[2] F. Leymann, D.Roller *Production Workflow - Concepts and Techniques*, Prentice-Hall Inc, 2000

[3] D. Gannon, E.Deelman, M. Shields, I.Taylor *Workflows for e-Science - Scientific Workflows for Grids*, Springer, 2007