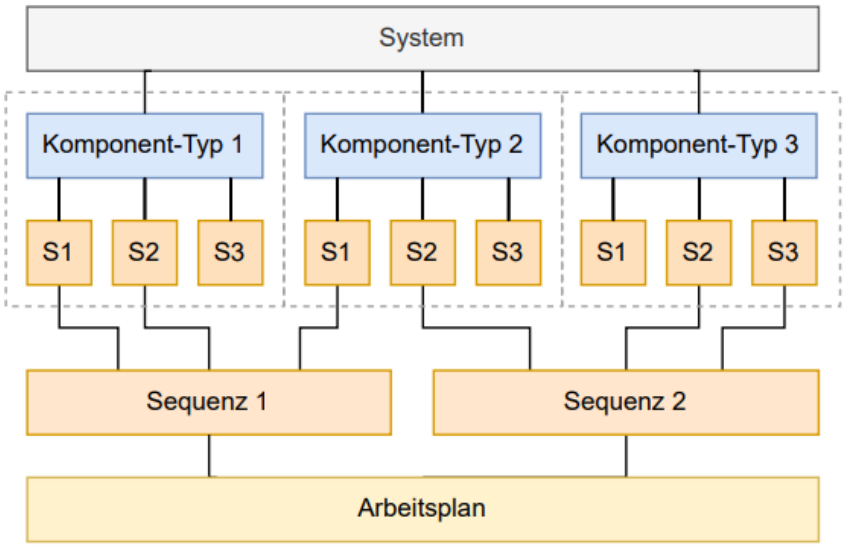
Arbeitspaket 5: Skills

Einführungstext

Kompetenzen von Skills

Für die Definition eines Skills wird nicht die Funktion des Gesamtsystem so weit wie möglich in Teilfunktionalitäten heruntergebrochen, sondern die Funktionalitäten der Komponenten im System. Dabei betrachtet man jeden Komponenten-Typ, so weit wie möglich, einzeln. Der Vorteil dieser Definition ist, dass für neue oder andere Systeme dieselben Skills verwendet werden können. Ein Roboter hat in jedem System die gleichen Grundfunktionalitäten und somit Skills. Der Skill ist Anwendungsunabhängig. Die aus den Skills erstellten Sequenzen bilden die Funktion der Anwendung ab. Damit unterscheidet sich die Definierung der Skills von vergleichbaren Projekten wie z.B. der bereits durchgeführten Master-Thesis auf Basis von ROS (Ref). Dort haben Skills Funktionalitäten von mehreren Komponenten ineinander kombiniert, wodurch der Skill stärker an das spezifische System gebunden war.



Die Kompetenzen eines Skills lassen sich in vier Bereiche aufteilen:

Zuweisung:	Der Skills ist für die Zuweisung einer Komponente zuständig. Es muss definiert werden können, welche Komponente den Skill ausführt.
Umsetzung:	Die definierte Grundfunktion muss innerhalb des Skills umgesetzt werden. Der Skill muss mittels Parameter-Inputs flexibel eingesetzt werden können.
Verarbeitung:	Die Informationen der Grundfunktion wird so ausgegeben, dass das Anlagenobjekt damit arbeiten und die reale Komponente ansteuern kann.
Auswertung:	Die momentane Situation der Komponente wird überwacht und ausgewertet. Der Skill kann auf bestimmte Situationen reagieren.

Definition von Anwendungs-Skills

Um die benötigten Skills für die Anwendung zu definieren, werden im ersten Schritt die allgemeinen Arbeitsschritte basierend auf dem mechanischen Aufbau (siehe Verweis) festgelegt. Dabei wird von der Ausgangssituation ausgegangen, dass alle Teile in ihren Lagerpositionen abgelegt sind, der Roboter sich in der Home-Position befindet und alle Komponenten eingeschaltet sowie betriebsbereit sind.

Schritt	Aktivität	Komponente
1	Position von Platte 1 in Lagerung erkennen	Kamerasystem
2	Platte 1 an Montageposition bringen und mittels L-Stück ausrichten	Roboter, Greifer, Kraftsensor
3	Position von Platte 2 in Lagerung erkennen	Kamerasystem
4	Platte 2 an Montageposition bringen und mit Platte 1 zusammenführen	Roboter, Greifer, Kraftsensor
5	Position der Befestigungslöcher in den Platten ermitteln	Kamerasystem
6	Position von Befestigungsblech in Lagerung erkennen	Kamerasystem
7	Befestigungsblech an Montageposition bringen	Roboter, Greifer, Kraftsensor
8	Position von Stift 1 in Lagerung erkennen	Kamerasystem
9	Stift 1 an korrekte Position bringen	Roboter, Greifer
10	Befestigungsblech mit Platte 1 verbinden, durch Eindrücken von Stift 1	Roboter, Kraftsensor
11	Wiederholen von Schritt 8 – 10 für Stift 2 bis 3	

Eine detaillierte Auflistung der Arbeitsschritte wird im Anhang beigefügt. Aus diesem lässt sich erkennen, dass sich diverse Schritte mit kleinen Anpassungen wiederholen. Diese sich wiederholenden Arbeitsschritte definierten die Skills. Ein entscheidender Aspekt dabei ist, dass der Roboter und der Kraftsensor als separate Komponenten betrachtet werden. Der Kraftsensor erweitert die Fähigkeiten des Roboters zwar und damit dessen Skills, jedoch kann der Roboter auch ohne Kraftsensor betrieben werden. Der Kraftsensor wird als eigene Objektklasse abgebildet, jedoch besitzt dieser keinen eigenen Skill. Folgende Skills wurden definiert, welche den Prozess abdecken.

Komponente	Skill	Bemerkung
Kamerasystem: *(Kamera + Vision)*	- Bild aufnehmen - Objekt erkennen - Greifposition ermitteln	
Roboter:	- Position anfahren - Kontrolliert bewegen	Die Zielposition wird angegeben Die Bewegung wird in Echtzeit vorgegeben und mit Sensor überwacht
Greifer: *(mit Sensoren)*	- Backenposition anfahren	Der Skill kann eine bestimmte Position anfahren. Der Greifer kann dadurch geöffnet oder geschlossen werden.

Die definierten Skills werden innerhalb der Umsetzung (Kapitelverweis) detaillierter beschrieben.

Definierung der Skill-Struktur

Alle Skills sollen mit der gleichen Struktur aufgebaut und jeweils nur mit den prozessspezifischen Funktionen ergänzt werden. Ein wichtiger Aspekt dieser Grundstruktur sind die Ein- und Ausgänge, welcher ein Skill minimal benötigt, um im Kontext der Systemstruktur funktionieren zu können. Auch Eigenschaften sind Teil der Grundstruktur, da Informationen auch über diese weitergegeben werden können. Ein Skill muss über die definierten Schnittstellen (Kapitel XXX) mit dem System und dem Objekt interagieren können.
Für die Definition der Ein- und Ausgangsvariablen wird PLCopen-Standard als Basis verwendet (Verweis). Dieser definiert wie ein Funktionsbaustein betätigt wird und welche Rückmeldungen dieser liefert. Ein Skill besitzt folgende Variablen und Eigenschaften.

Eingangsvariablen:

Nr	Variable	Typ	Beschreibung
E1	bExecute	BOOL	Trigger für Ausführung von Skill
E2	eSysCommand	eSystemCommand	Befehlsvariabel von System zu Skill
E3	eObjState	eObjectState	Informationen über Zustand von Objekt (Anlagenmodell)
E4	eSysState	eSystemState	Informationen über System (Systemparameter)

Die Variabel E1 wird für das Starten des Skills verwendet. Der Skill soll auf die steigende Flanke reagieren und kann erst wieder gestartet werden, wenn dieser deaktiviert und wieder neu aktiviert wird. Variabel E2 bis E4 werden als benutzerdefinierter Datentyp umgesetzt, welche die verschiedenen Befehl- oder Zustandsoptionen wiederspiegeln.

Listen-Nummer	eSystemCommand	eObjectState	eSystemState
0	KEINE	AUS	AUS
1	AUSSCHALTEN	BEREIT	BEREIT
2	EINSCHALTEN	MANUELL	LAUFEND
3	STOPPEN	LAUFEND	GESTOPPT
4	RESETTEN	ABGESCHLOSSEN_INTERN	FEHLER
5	/	ABGESCHLOSSEN_EXTERN	/
6	/	GESTOPPT	/
7	/	FEHLER	/

Ausgangsvariablen:

Nr	Variable	Typ	Beschreibung
A1	bDone	BOOL	Information ob Skill erfolgreich ausgeführt wurde
A2	bBusy	BOOL	Information ob Skill im Moment ausgeführt wird
A3	bLimit	BOOL	Information ob Skill an einem Limit angekommen ist
A4	bError	BOOL	Information ob sich Skill im Moment im Fehlerzustand befindet
A5	iErrorID	INT	Information um welchen Fehler es sich handelt
A6	eSkillCommand	eSkillCommand	Befehlsvariabel von Skill zu für Objekte (Anlagenmodell).

Die Variable A3 gibt an, ob der Skill ein definiertes Limit erreicht hat. Dies kann z.B. eine Kraft- oder Zeitvorgabe sein. Der Skill gibt dabei keinen Fehler an. Die Idee ist, dass der Ablauf auf diese Information reagieren kann, um eine Korrektur vornehmen zu können.

Die Variabel A6 wurde mit einem benutzerdefinierter Datentyp umgesetzt, welcher die Befehle des Skills and das Objekt definiert.

Listen-Nummer	eSkillCommand
0	KEINE
1	STARTEN
2	STOPPEN
3	RESETTEN

Eigenschaften (Property)

Nr	Variable	Typ	Beschreibung
P1	P_State (GET)	eSkillState	Information über Zustand von Skill

Die Eigenschaft P1 wurde mit einem benutzerdefinierter Datentyp umgesetzt, welcher die Zustände des Skills abbildet.

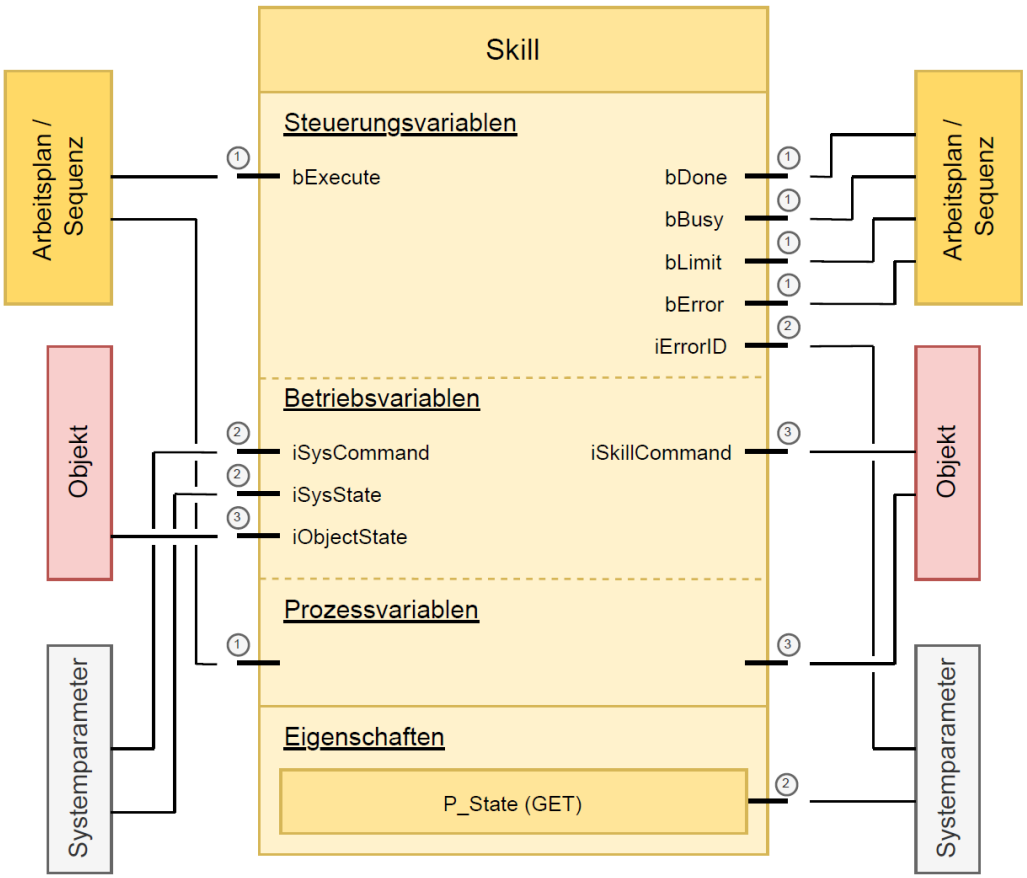
Listen-Nummer	eSkillState
0	BEREIT
1	LAUFEND
2	ABGESCHLOSSEN
3	FEHLER
4	LIMITE
5	ERREICHT

Zusammen mit der allgemeinen Struktur der Software (Kapitel XXX) kann nun der Grundaufbau eines Skills und seine Schnittstellen komplett definiert werden.

Die Variablen lassen sich in drei Kategorien aufteilen. Die Steuerungsvariablen wurden durch den PLCopen-Standard vorgegeben. Diese Variablen werden innerhalb des Prozessmodells (Arbeitsplan / Sequenzen) verwendet um den Skill auszuführen und auf diesen, bezogen auf den Ablauf, zu reagieren. Die einzige Ausnahme bildet die Variabel «iErrorID». Diese ist Teil der Koordinationsschnittstelle und gibt die Information, um welchen Fehler es sich handelt, an die Systemparameter weiter.

Die Betriebsvariablen sind für den eigentlich Betrieb des Skills verantwortlich. Dies umfasst die Befehlsvariablen «iSysCommand» und «iSkillCommand», wie auch die Zustandsvariablen «iSysState» und «iObjectState». Die Interaktion mit den Systemparameter ist Teil der Koordinationsschnittstelle und die Objektinteraktion Teil der Modellschnittstelle.

Die letzte Variabel-Kategorie sind die Prozessvariablen. Dies umfasst prozessspezifische Informationen, welche für das Ausführen des Skills relevant sind. Diese sind entsprechend von Skill zu Skill unterschiedlich. Vorgegeben werden diese über den Arbeitsplan oder Sequenz (Interne Prozessmodellschnittstelle). Die Informationen werden vom Skill an das Objekt weitergegeben (Modellschnittstelle).



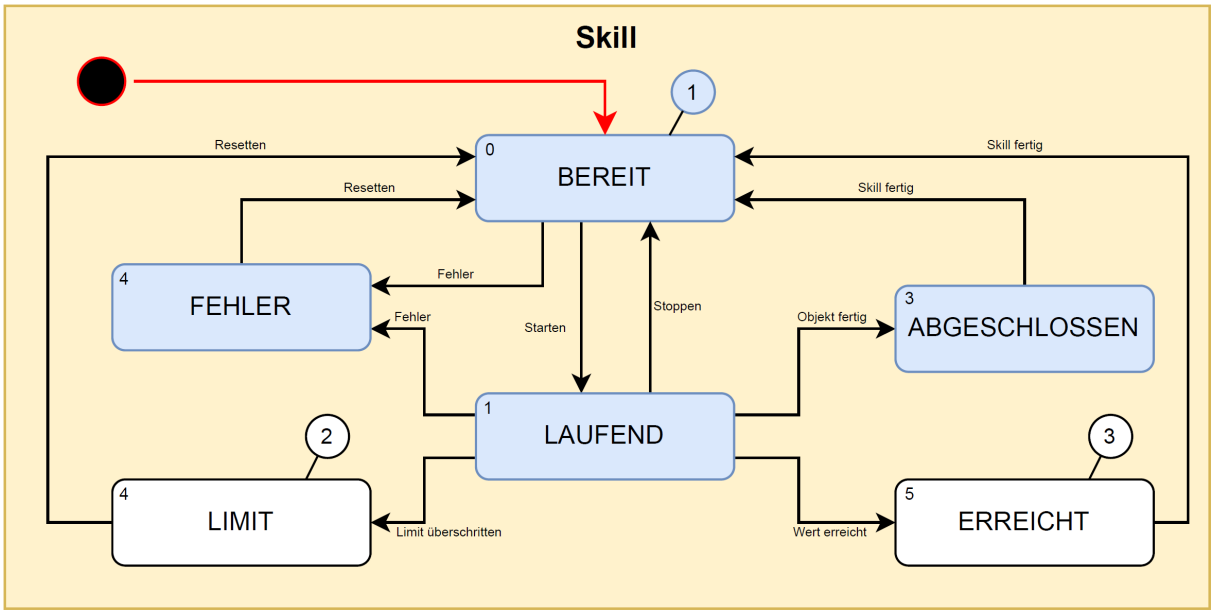
1: Interne Schnittstelle von Prozessmodell

2: Koordinationsschnittstelle

3: Modellschnittstelle

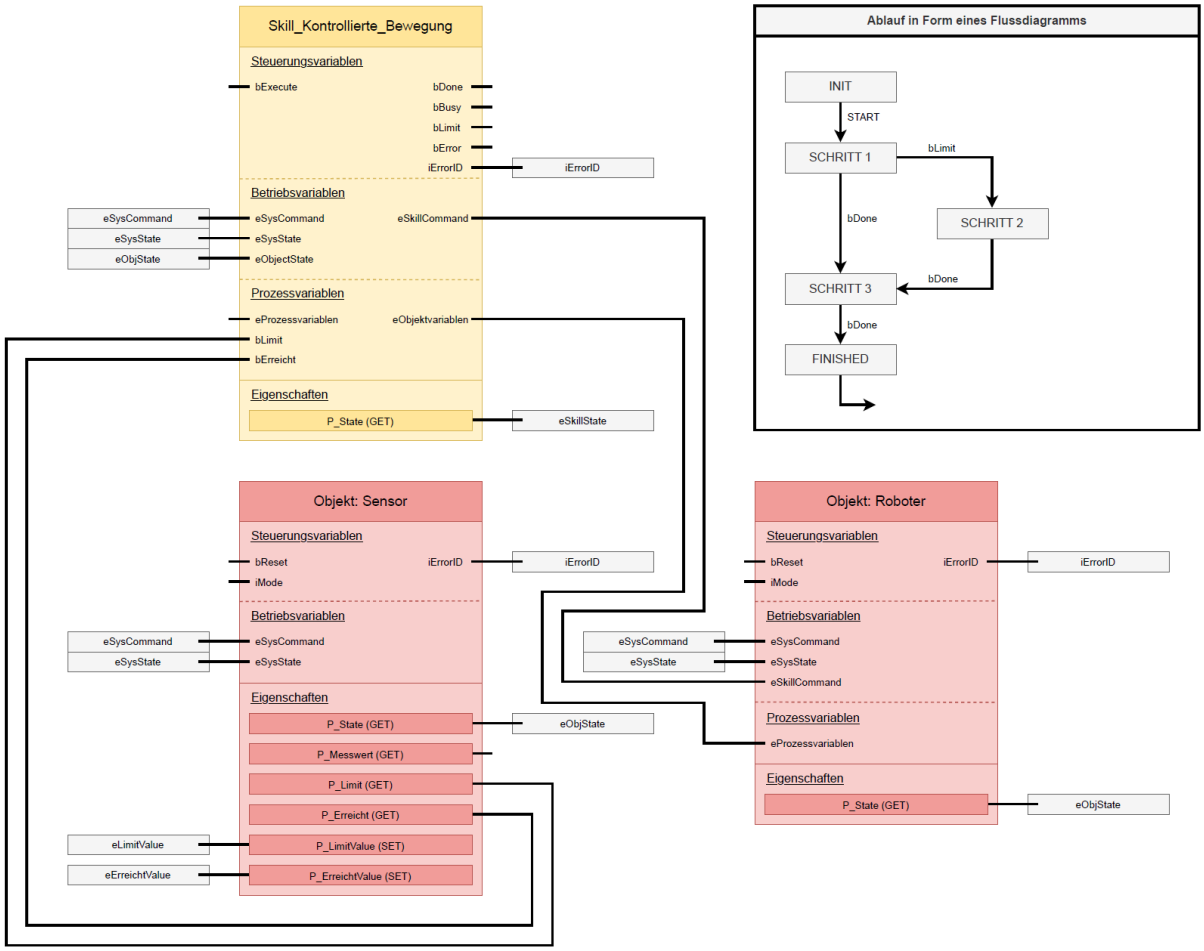
Konfiguration eines Skills

Nicht jeder Skill benötigt alle Zustände. Damit der Skill so übersichtlich wie möglich bleit, sollen auch nur die Zustände verwendet werden, welche benötigt werden. Ein Skill kann drei Konfigurationen einnehmen.



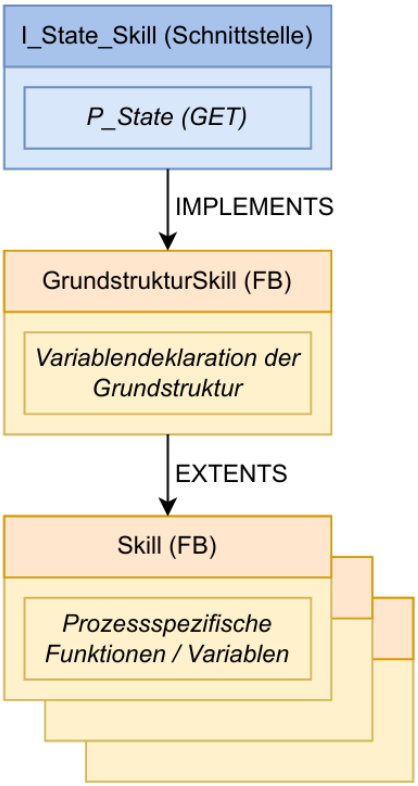
Konfiguration 1:	Stellt die Grundkonfiguration dar. Jeder Skill muss diese Zustände besitzen. Hierbei handelt es sich um Skills, welche nur durch das Objekt abgeschlossen werden, z.B. eine einfache Punkt-Zu-Punkt-Bewegung des Roboters.
Konfiguration 2:	Bei dieser Konfiguration kommt der LIMIT-Zustand dazu. Dieser wird benötigt, wenn es eine Limit-Bedingung gibt, z.B. einen Grenzwert für die Kraft oder eine maximale Zeitdauer.
Konfiguration 3:	Bei der letzten Konfiguration wird der ERREICHT-Zustand ergänzt. Dieser gibt an, ob ein definiertes Ziel erreicht wurde, dies kann z.B. eine Kraft sein.

Skills mit Reaktion auf Feedback



Umsetzung von Skills

Die Skill-Struktur gibt vor, dass alle Skills einen identischen Grundaufbau besitzen. Um dies zu realisieren, wird mit «Schnittstellen» und «Vererbungen» gearbeitet. Mit Schnittstellen können Methoden und Eigenschaften vorgeschrieben werden, welche zwingend umgesetzt werden müssen. Jeder Skill besitzt eine Eigenschaft zur Ausgabe des aktuellen Standes (P_State). Diese Eigenschaft wird im Funktionsbaustein «GrundstrukturSkill» implementiert. In diesem Baustein werden alle Variablen definiert, welche alle Skills besitzen müssen. Dies umfasst die bereits definierten Ein- und Ausgangsvariablen, wie auch die internen Variablen. Die unterschiedlichen Skills können über eine Vererbung «EXTENDS» auf diese Variablen zugreifen. Die Vererbung stellt die Variablen separat für jeden Skill zur Verfügung. Ein Skill kann nicht die Variablen eines anderen Skills verändern. Durch diese Struktur wird der Aufbau eines Skill deutlich vereinfacht. Dieser wird nun noch mit Prozessspezifischen Variablen, Methoden und Eigenschaften versehen.



Neben den definierten Ein- und Ausgangsvariablen besitzen alle Skills die folgenden internen Variablen:

Variable	Typ	Beschreibung
Managementvariablen		
iState	eSkillState	Informationen über Zustand von Skill
bTargetReached	BOOL	Information ob Prozessziel erreicht wurde
bLimitReached	BOOL	Information ob definiertes Limit erreicht wurde
bGestartet	BOOL	
bGestoppt	BOOL	
Startvariablen		
fTrigger	R_TRIG	Erkennung einer steigenden Flanke für den Start des Skills
fSwitch	SR	Setzten und zurücksetzen des Signals
Zustandsvariablen		
bStarten	BOOL	Transition-Variable von BEREIT zu LAUFEN
bStoppen	BOOL	Transition-Variable von LAUFEND zu BEREIT
bObjektFertig	BOOL	Transition-Variable von LAUFEND zu ABGESCHLOSSEN
bWertErreicht	BOOL	Transition-Variable von LAUFEND zu ERREICHT
bLimitErreicht	BOOL	Transition-Variable von LAUFEND zu LIMIT
bSkillFertig	BOOL	Transition-Variable von ABGESCHLOSSEN / ERREICHT zu BEREIT
bFehler	BOOL	Transition-Variable von BEREIT / LAUFEND zu FEHLER
bResetten	BOOL	Transition-Variable von FEHLER / LIMIT zu BEREIT