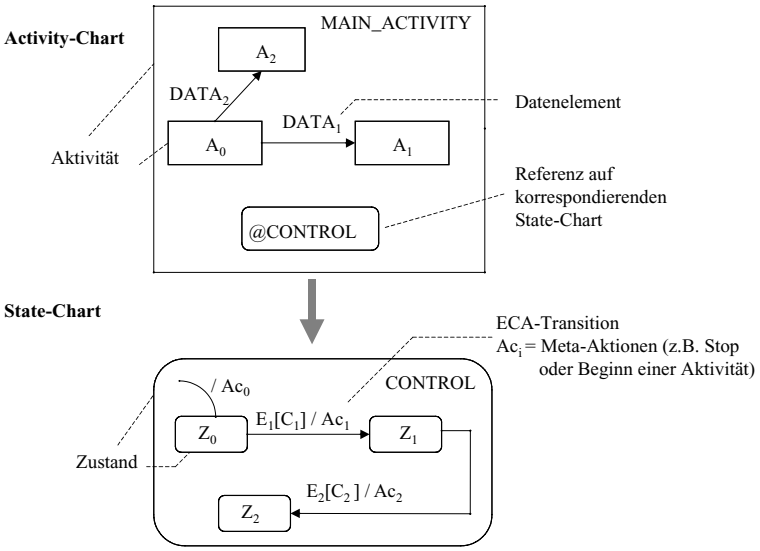
State- und Activity-Charts: Hauptmerkmale

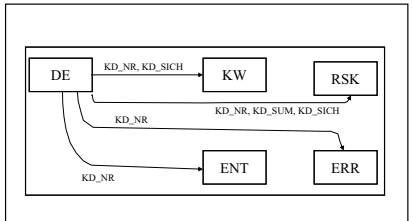
- Spezifikationssprache für das Verhalten aktiver Systeme
- Basieren auf einem Formalismus von Harel & Pnueli, 1987
- Activity-Charts
  - Hierarchische Zerlegung eines Systems in Funktionseinheiten (Activities)
  - Beschreibung des Datenflusses zwischen Aktivitäten
  - Typische Aktivitäten: Programmaufrufe, Datenbank-Operationen, Meldungen an Benutzer
  - Zu jedem Activity-Chart *AC* korrespondiert ein State-Charts, welches das „Verhalten“ von *AC* beschreibt
- State-Charts
  - Spezifikation des Kontrollflusses zwischen Aktivitäten
  - Zuordnung von ECA-Regeln zu Transitionen
- Semantik einer Transition $X \rightarrow Y$ mit Beschriftung *E[C]/A*:
  - Falls *X* erreicht, Ereignis *E* eingetreten sowie die Bedingung *C* bzgl. *E* erfüllt ist: „Feuern“ von $X \rightarrow Y$
  - Feuern heisst: Aktivität *A* wird ausgeführt, und der Zustand *Y* wird betreten
- Trennung zwischen Kontroll- und Datenfluss (im Gegensatz zu Petri-Netzen)

- Module-Charts für die Gruppierung von Activity-Charts zu größeren Funktions-Einheiten
- Analyse-Werkzeuge
- Übersichtliche graphische Notation

State- und Activity-Charts: Schematische Übersicht



State- und Activity-Charts: Beispiel



DE - Dateneingabe

KW- Kreditwürdigkeitsprüfung

RSK - Risikoevaluation

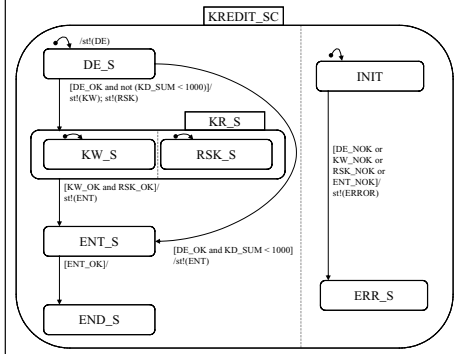
ENT - Entscheidung

ERR - Fehlerbehandlung

KD_NR - Kunden-Nummer,

KD_SICH - Kreditsicherheiten

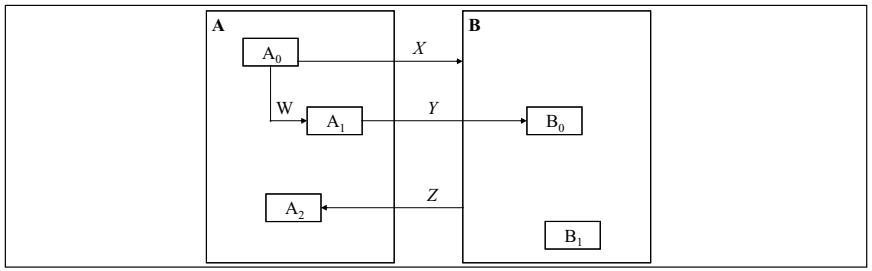
KD_SUM - Kreditsumme



Activity-Charts: Details

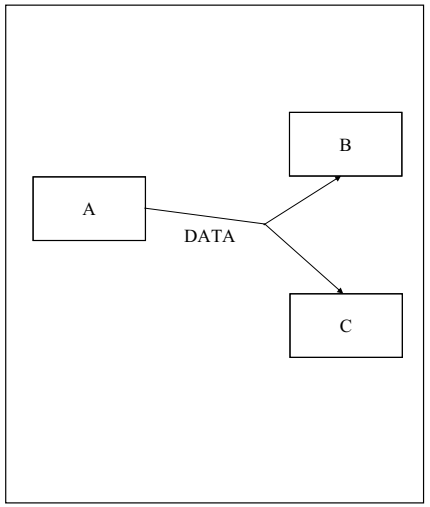
- Beschreibung der funktionalen Sicht eines Systems
- Hierarchische Datenflussdiagramme
- Spezifikation des Datenflusses zwischen Aktivitäten
  - Festlegung, *welche* Daten zwischen Aktivitäten fließen
  - Keine Detail-Angaben über Zeitpunkt und Häufigkeit des Datenaustauschs zwischen Aktivitäten
- Ein Activity-Chart macht bzgl. der Aktivitäten keine Aussage über
  - Ausführungsreihenfolge
  - Ausführungshäufigkeit
  - Nebenläufigkeiten
  - Dauer oder Verzögerungszeiten
- Referenzierung des steuernden State-Charts durch Symbol @CONTROL

Activity-Charts: Datenflussmodellierung



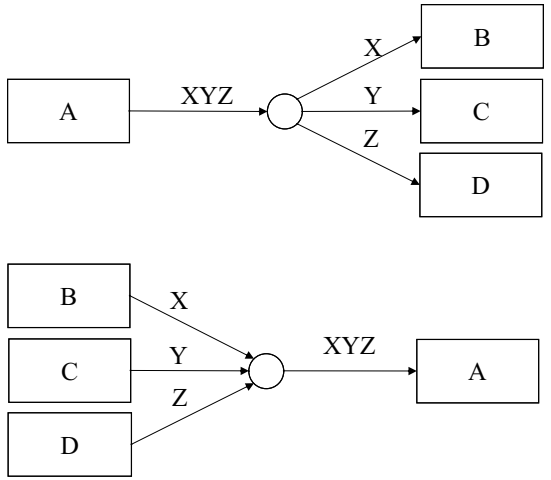
- Activity Chart *A* umfaßt die drei Subaktivitäten A_0, A_1 und A_2, Activity Chart *B* umfaßt B_0 und B_1
- Datenflüsse im Beispiel u.a.:
  - *W* von A_0 nach A_1, *X* von A_0 nach *B*
  - *Y* von A_1 nach B_0, *Z* von *B* nach A_2
  - Datum *X* steht sowohl *B* als auch B_0 und B_1 zur Verfügung
  - Datum *Z* kann sowohl von *B* als auch von B_0 und B_1 manipuliert worden sein

Activity-Charts: Verbindungen (Joint Connectors)

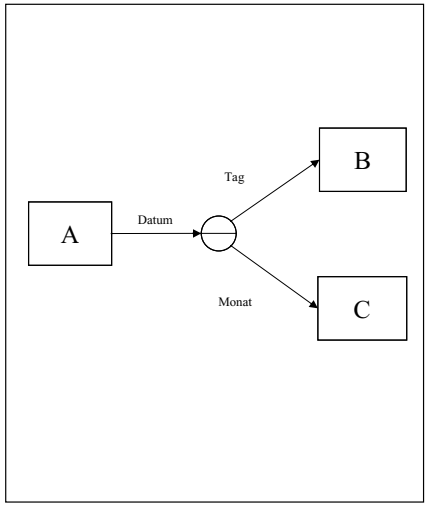


Datum *DATA* wird sowohl an *B* als auch an *C* geliefert

Activity-Charts: Zerlegung und Zusammenführung (Junction Connectors)



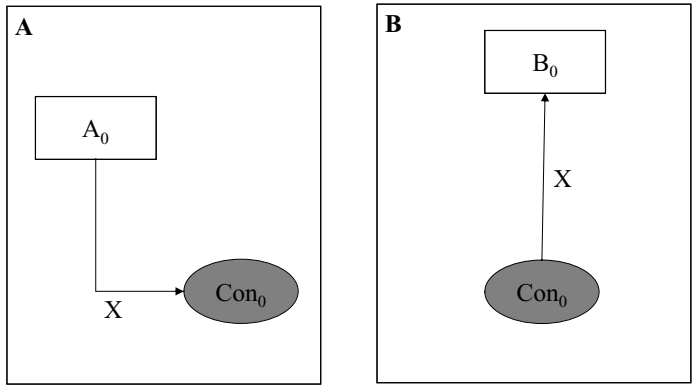
Activity-Charts: Record-Zerlegung (Record Connector)



- Zerlegung eines Records in seine Komponenten
- Analog zum *Junction Connector* auch *Zusammenführung* zu einem Record möglich

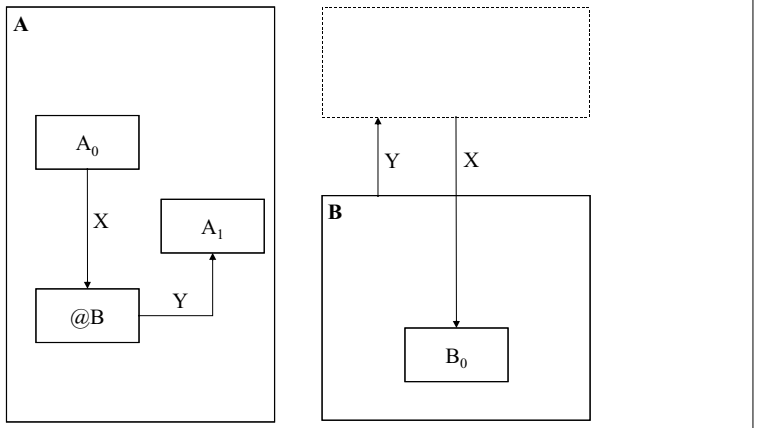
Activity-Charts: Chart-Verbindungen (Diagram Connectors)

- Ermöglichen Datenflüsse zwischen separat modellierten Activity-Charts (Information Hiding)



Activity-Charts: Matching Flows

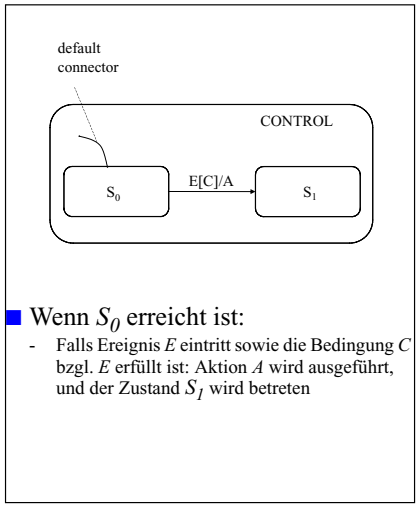
- Gleiche Funktionalität wie Diagram Connectors
- Repräsentierung externer Aktivitäten durch gestrichelte Kästchen



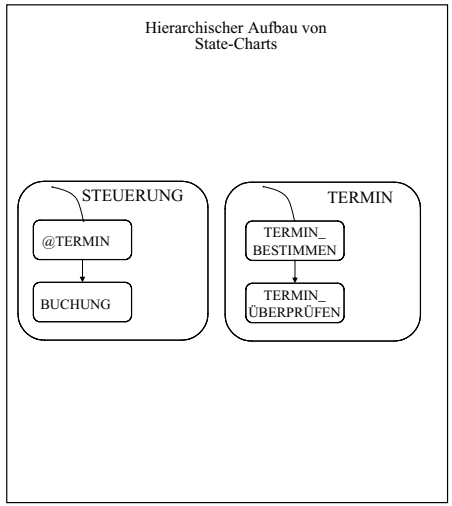
State-Charts: Details

- Eine State-Chart beschreibt das *Verhalten* (d.h. den Kontrollfluß) eines Systems bzw. des ihm zugeordneten Activity-Charts
- Graphische Darstellung als Zustandsgraph mit Transitionen
- Basieren auf erweiterten endlichen Automaten
- Wesentliche Ergänzungen:
  - Verfeinerung von Zuständen durch hierarchische Zerlegung in Subautomaten
  - Nebenläufigkeit
  - nicht-deterministische Transitionswahl
  - Verknüpfung mit getrennt beschriebenen Aktivitäten (→ Activity-Chart)
- Kontrollfluß wird bei Zustands-Transitionen u.a. durch das Starten und Beenden von Aktivitäten gesteuert
- Graphische Konvention: Rechteck mit abgerundeten Ecken

State-Charts: Schematische Übersicht

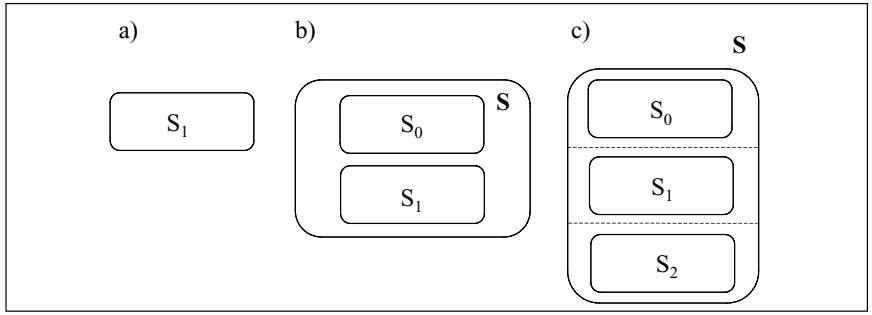


- Wenn S_0 erreicht ist:
  - Falls Ereignis E eintritt sowie die Bedingung C bzgl. E erfüllt ist: Aktion A wird ausgeführt, und der Zustand S_1 wird betreten



State-Charts: Zustandstypen

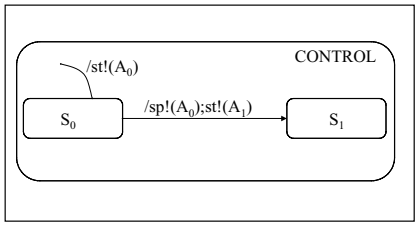
- Elementar: einfacher Zustand, ohne weitere Verfeinerung (Abb. a)
- ODER-Zustände: exklusive Zustände, d.h. das System kann sich in maximal einem der Subzustände befinden → Modellierung bedingter Ausführungszweige (Abb. b)

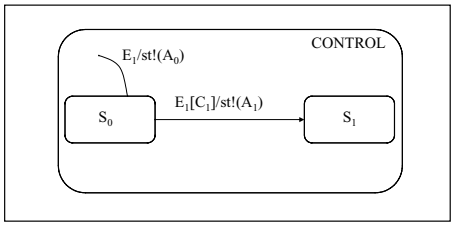


State-Charts: ECA-Übergänge

- Anweisungen zur Steuerung von Aktivitäten:

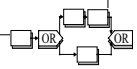
Anweisung	Kurzform	Bedeutung
start!(A)	st!	starte Aktivität *A*
stop!(A)	sp!	beendige Aktivität *A*
suspend!(A)	sd!	setze Aktivität *A* aus
resume!(A)	rs!	setze die Ausführung von Aktivität *A* fort
schedule!(A,time)	sc!	starte *A* in *time* Zeiteinheiten



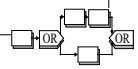


Fallbeispiel: State-Activity-Chart für *Klärung der Vorbedingungen*

Fallbeispiel: State-Activity-Chart für *Bearbeitung SHK-Antrag*



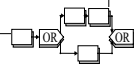
Fallbeispiel: State-Activity-Chart für *Bearbeitung Hauptvertrag*



State- und Activity-Charts: Zusammenfassung

- Spezifikationsprache für das Verhalten von aktiven Systemen; geeignet für Workflow-Spezifikation
- Dualer Spezifikationsansatz
  - Activity-Charts: Datenflußbeschreibung, hierarchische Zerlegung von Aktivitäten, funktionale Sicht
  - State-Charts: Kontrollfluß, bedingte Zustandstransitionen
- Vorteile
  - Theoretische Fundierung
  - Trennung von Daten- und Kontrollfluss, Übersichtlichkeit durch hierarchische Zerlegung
  - gute Fehlererkennung durch Analysewerkzeuge
- Nachteile
  - „Unübersichtlichkeit“ der State-Charts: Aktivitätssteuerung in den Transitionen „versteckt“
  - Limitierte temporale Unterstützung

- Produkte
  - Statemate (I-Logixs); http://www.ilogix.com/
  - ObjChart (IBM); http://www.software.ibm.com/
- Forschungsgruppen u.a.
  - Weikum et al. (Universität Saarbrücken; http://www-dbs.cs.uni-sb.de/): Einsatz von State-Activity-Charts im Workflow-Projekt MENTOR



Statemate (I-Logixs): Screenshot