



Message Sequence Charts

Definition und Übersetzung für
Petrietze

Ulrik Schrimpf



Gliederung

1. **Ursprung und Motivation**
2. **Basic MSC's**
3. **Strukturelle Sprachkonstrukte und High-Level-MSCs**
4. **Textuelle Darstellung**
5. **Übersetzung in Petrinetze**
6. **Quellen**



1. Ursprung und Motivation

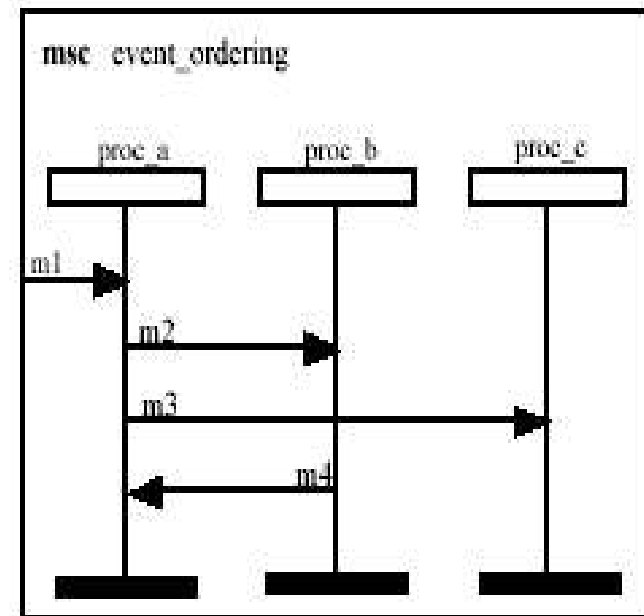
- **Geschichte:**
 - **Verwendung schon vor Standardisierung (OSI Time Sequence Diagram)**
 - **1990 Standardisierung durch ITU**
 - **1993 Standard Z.120**
 - **1995 Ergänzung um formale Semantik**
 - **1996 MSC'96**
 - **2000 MSC-2000**



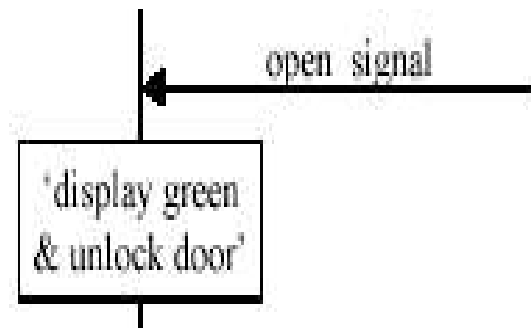
1. Ursprung und Motivation

- **Darstellung einzelner Systemabläufe und der Kommunikation verschiedener Prozesse**
- **Zeitliche Ordnung von Interaktion**
- **Anwendung:**
 - **Anforderungen**
 - **Dokumentation**
 - **Testfälle**
- **MSC's liegen in textueller(MSC/PR) und graphischer(MSC/GR) Form vor**
- **nahe verwandt mit SDL**
- **Sequence Diagramm in UML sind Variante**

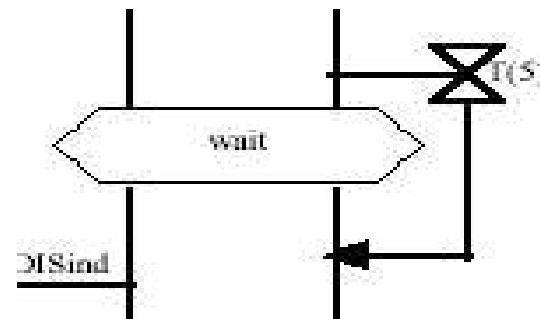
- **Instanzen & Nachrichten**
 - Instanzen sind Komponenten die Nachrichten austauschen
 - Instanz als vertikale Linie
 - Nachricht als horizontale und geneigte Pfeile
- **Systemumgebung**
 - Diagrammrahmen um MSC
 - Nachrichten herein/hinaus beginnen/enden in der Systemumgebung
 - Für Komposition wichtig



- **Aktion**
 - Rechtecksymbol
 - Beschreibt Aktion, die von Instanz ausgeführt wird



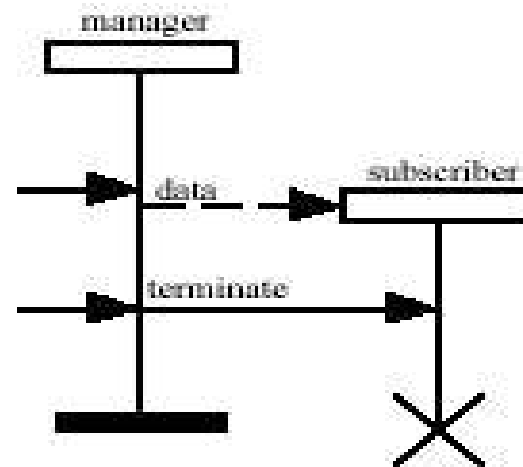
- **Timer**
 - Start, Stop und Timeout
 - Instanz zugeordnet
 - Name und Parameter können zugeordnet werden





2. Basic MSC's

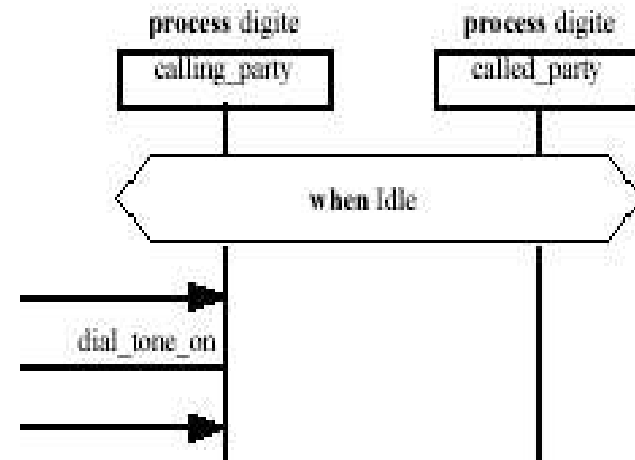
- **Create & Stop**
 - Dynamisches Erschaffen und Beenden von Instanzen
 - Create als gestrichelter Pfeil mit Parametern
 - Stop als Kreuz am Ende der Instanz





2. Basic MSC's

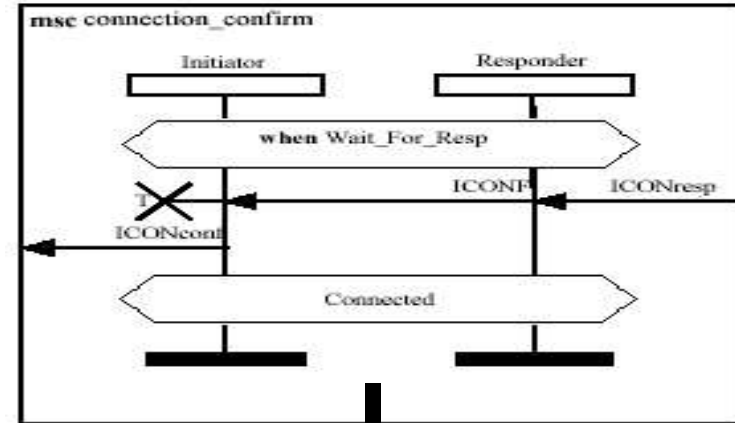
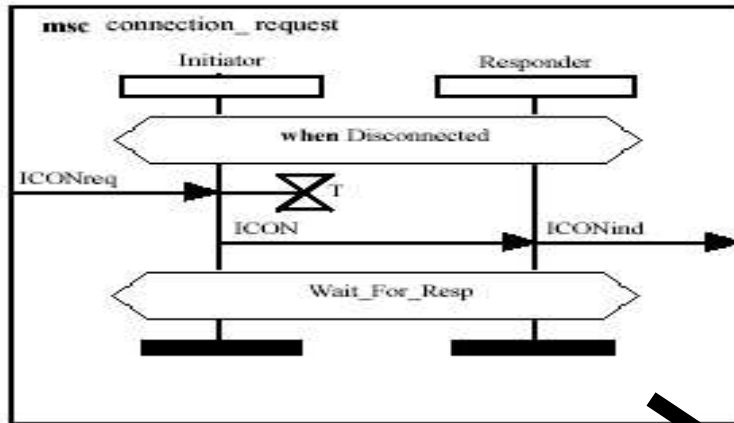
- **Bedingung**
 - Global und lokal
 - Zustände, die sich auf Instanzen beziehen
 - Sechsecke
 - Ermöglichen Komposition und Dekomposition



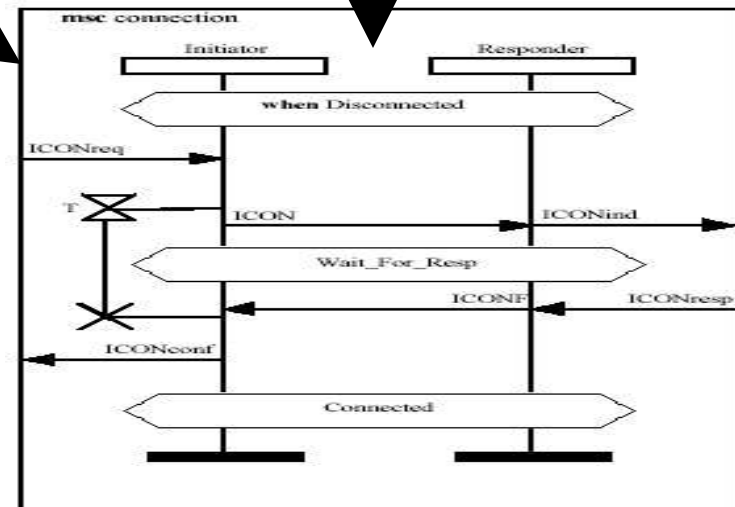


2. Basic MSC's

- **Kausale Ordnung als gestrichelte Linie mit Pfeil der Mitte**
- **Coregionen als gepunktete Linie in Instanzen, innerhalb dieser sind Ereignisse nicht geordnet**
- **SUBMSC's verfeinern MSC's**
- **Formale Semantik ordnet Kommunikationsereignisse auf Instanzen**



- **Komposition & Dekomposition**
 - Parallel und sequentiell





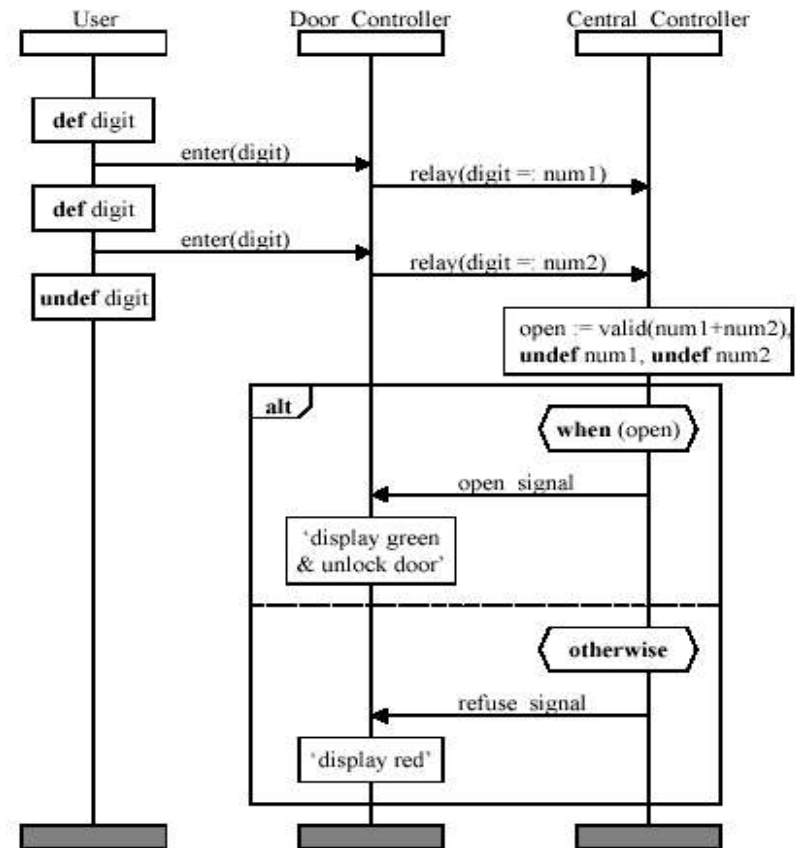
3. Strukturelle Sprachkonstrukte und High-Level-MSD's

- **Erweiterung für**
 - **Kombination komplexer Abläufe**
 - **Wiederverwendung**
 - **Verfeinerung**
 - **Definition von Ereignisstrukturen**



3. Strukturelle Sprachkonstrukte und High-Level-MSC's

- **InlineExpression**
 - alt (alternative Teilabläufe)
 - par (parallele T.)
 - loop (Wiederholung)
 - opt (optionale T.)
 - exc (Ausnahmen)
 - Operator in linker oberer Ecke

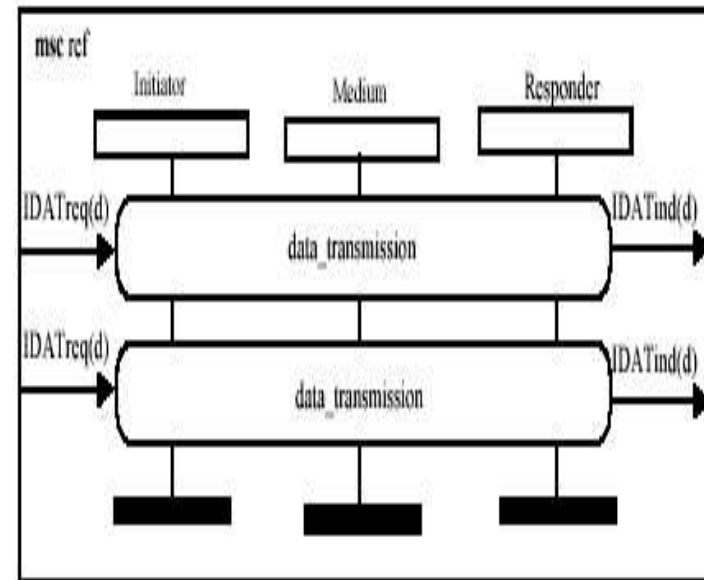




3. Strukturelle Sprachkonstrukte und High-Level-MSD's

- **Referenzen**

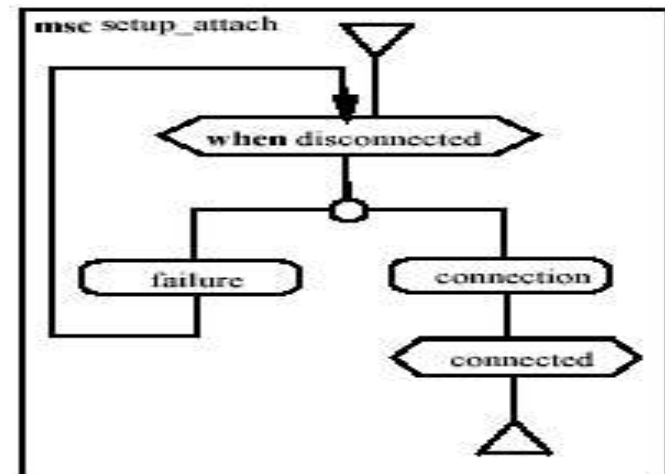
- **MSC's in MSC's**
- **Referenziert über den Namen**
- **Rechteck mit abgerundeten Ecken**





3. Strukturelle Sprachkonstrukte und High-Level-MSCs

- **High-Level-MSCs**
 - Kombination von MSC's in Form von gerichteten Graphen
 - Abstraktion von Instanzen und Messages
 - Auch Roadmaps
 - Knoten sind:
 - Anfangsknoten
 - Endknoten
 - Konnektoren
 - Referenzen
 - Bedingungen





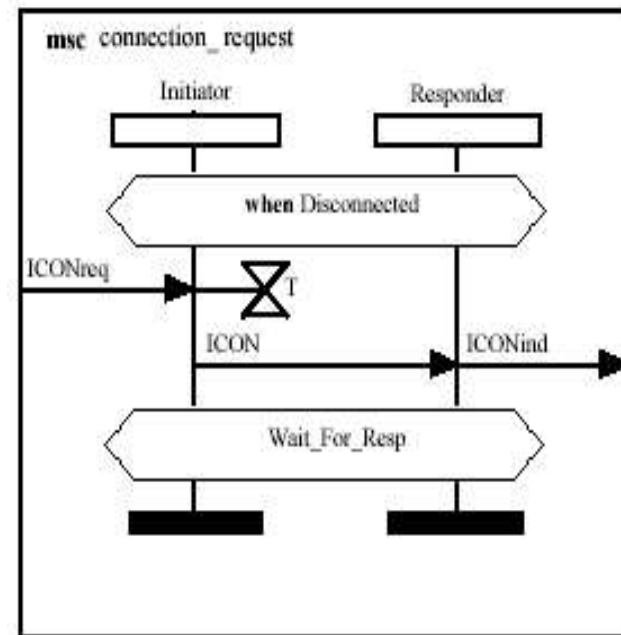
4. Textuelle Darstellung

- **Instanzenorientierte Syntax**
 - Nach Instanzen geordnet
- **Ereignisorientierte Syntax**
 - Näher mit grafischen Variante verwandt
 - Transformation einfacher
 - Nachrichtenerreignisse zeitlich geordnet



4. Textuelle Darstellung

```
msc connection_request;  
inst Initiator;  
inst Responder;  
gate out ICONreq to Initiator;  
gate in ICONind from Responder;  
instance Initiator;  
condition when Disconnected shared all;  
in ICONreq from env;  
starttimer T;  
out ICON to Responder;  
condition Wait_For_Resp shared all;  
endinstance;  
instance Responder;  
condition when Disconnected shared all;  
in ICON from Initiator;  
out ICONind to env;  
condition Wait_For_Resp shared all;  
endinstance;  
endmsc;
```





5. Übersetzung in Petrinetze

- **Vereinfachung in:**
 - **MSC**
 - **Instanz**
 - **Ereignisse**
 - **Coregionen**
- **Komponenten besitzen Eingang, Ausgang und Ausführung von Aktionen**
- **Überführte HMSC's besitzen zudem die Möglichkeit alternative Aktionen auszuführen**

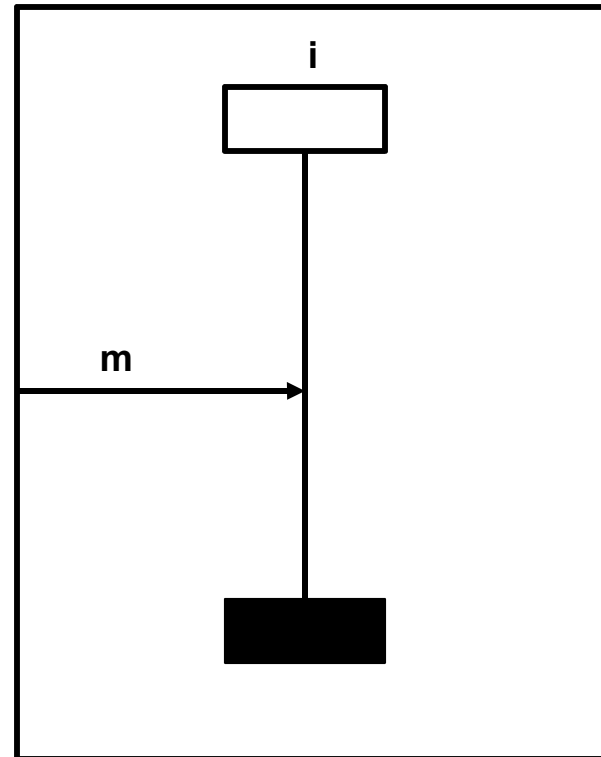
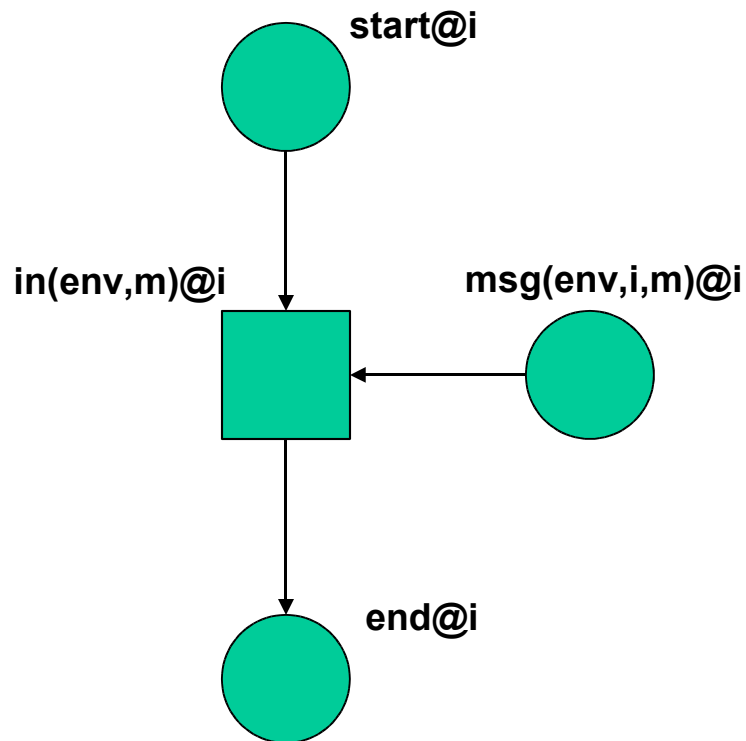


5.Übersetzung in Petrinetze

- Als Beschriftungen werden sie Sätze InstID, MscID, MsgID und ActID gewählt
- Beschriftungen werden aus diesen Sätzen zusammengestellt
- Beispiel: $msg(i,j,m)$ ist die Beschriftung für eine Stelle die eine Nachricht m , die von der Instanz i zu der Instanz j gesendet wird
- Die Beschriftungen $start@i$ und $end@i$ bezeichnen den Beginn und das Ende einer Instanz
- Transitionen werden mit *in* und *out* Beschriftungen versehen



5. Übersetzung in Petrinetze





6. Quellen

- **ITU-TS. *ITU-TS Recommendation Z.120: Message Sequence Chart (MSC)*. ITU-TS, Geneva, 1996**
- **S. Heymer. *A Semantics for MSC based on Petri-Net Components*. University of Lübeck, Institut of Telematics, 2000**
- **http://goethe.ira.uka.de/seiten/sys_spec/sys_msc4.pdf**
- **http://www.swe.informatik-goettingen.de/publications/MSC-Teil1/at0112_A19.pdf**