

# OWN COMMUNICATION MANAGEMENT

## Kodningsmanual v1.0

Jens Allwood, Elisabeth Ahlsen, Joakim Nivre, Staffan Larsson

October 10, 1997

## 1 Inledning

Syftet med detta dokument är att formulera principer för kodning av *own communication management* (OCM). OCM är ett samlingsnamn på de processer som talare använder för att reglera sina egna språkliga bidrag till kommunikativ interaktion (t.ex. planeringsfenomen, reparationer, editerande, självrättningar osv). Man kan urskilja två beskrivningsdimensioner för dessa processer: en relaterad till processens funktion och en baserad på uttrycksstruktur. Notera dock att det inte finns en klar skiljelinje mellan funktion och uttrycksstruktur; det vi kallar en ytstrukturell beskrivning är inte nödvändigtvis hundra procentigt ytstrukturorienterad, utan snarare *huvudsakligen* orienterad mot ytstruktur.

Exemplen i denna kodningsmanual är återgivna enligt transkriptionsstandarderna specificerade i [3] och [2].

## 2 Datorverktyg

Kodningen underlättas avsevärt av datoriserade transkriptionskodningsverktyg. Ett sådant verktyg är TRACTOR (Transcription Coding Tool). Detta verktyg och dess handhavande beskrivs i ett separat dokument ([1]), och den föreliggande kodningsmanualen är i stort sett oberoende av hur den praktiska kodningsprocessen utformas. Det bör noteras att TRACTOR-versionen av OCM-kodningsschemat använder sig av de engelska kodningsettiketterna.

## 3 Funktionskodning

OCM-fenomen kan alltså klassificeras både från ett funktionellt perspektiv och från ett perspektiv som tar hänsyn till uttryckets struktur. Vi kan skilja på två typer av OCM-funktioner:

1. Valrelaterad OCM (**choice**) hjälper talaren att vinna tid för processer som har att göra med fortsatt val av innehåll och typer av strukturerade uttryck, speciellt minnessökande, tvekan, planering.

**Exempel:** de e en va heter de valkyria

2. Ändringsrelaterad OCM (**change**) hjälper talaren att (på basis av diverse inre och yttre återkopplings-processer) ändra redan producerat innehåll, struktur eller uttryck. Exempel på ändringsrelaterat OCM är självreparationer och självrättningar.

**Exempel:** de e en blå ja1 menar röd bil

Kodning av dessa funktioner görs enligt följande procedur:

1. Märk ut den maximala<sup>1</sup> sekvens av ord som används för val- eller ändringsrelaterad OCM. (Denna sekvens motsvaras i exemplen ovan

---

<sup>1</sup>Att sekvensen ska vara maximal innebär att man istället för att koda en sammanhängande sekvens med flera förekomster av en och samma OCM-funktion bör koda

av text i fetstil.)

## 2. Välj motsvarande kod.

Observera att det ibland kan vara svårt att avgöra om ett segment ska märkas med **change** eller **choice**, t ex när en talare först förefaller söka efter ett ord (**choice**) för att sedan ge upp och istället börjar fundera på hur han/hon ska ändra på det som redan sagts (**change**). I sådana fall få kodaren helt enkelt lita till sina intuitioner och koda det första segmentet som **change** och det andra som **choice**.

I andra fall kan det vara motiverat att koda ett och samma segment som både **choice** och **change**, speciellt då ett och samma segment innehåller fler än en OCM-struktur (se kapitel 4).

**Exempel:** ja vill ha banan  $[[\text{nä:}]_{\text{simple\_OCM\_expr}}]_{\text{lengthening}}$  jordgubbsglass<sup>2</sup>

I detta fall bör **nä** kodas som både **choice** och **change** eftersom **nä** signalerar att talaren vill ändra på det redan sagda, och förlängningen av **ä** indikerar att talaren vill vinna tid för att välja hur han/hon ska fortsätta. Observera att kodning av båda funktionskategorierna **inte** får användas som ett sätt att indikera osäkerhet på vilken OCM-funktion ett segment har. I tveksamma fall måste kodaren bestämma sig för vad som verkar vara den dominerande funktionen och koda endast denna.

## 4 Uttrycksstruktur

Ett antal olika uttryck och operationer kan användas för att realisera OCM-funktioner. Många av dessa strukturer kan även användas i andra syften, men för OCM-kodning är vi bara intresserade av OCM-relaterade förekomster.

Bland de uttrycksstrukturer som realiserar OCM kan vi skilja på grundläggande och komplexa OCM-strukturer, där de senare är kombinationer av de förra.

---

hela sekvensen som en enda förekomst av OCM-funktionen.

**Exempel:** de e en  $[\text{ä0}]_{\text{choice}}$   $[/][_{\text{choice}}$  blå bil  $\rightarrow$  de e en  $[\text{ä0} //]_{\text{choice}}$  blå bil

<sup>2</sup>Detta exempel är konstruerat.

## 4.1 Grundläggande OCM-strukturer (basic\_ocm\_features)

- Grundläggande OCM-uttryck (basic\_ocm\_expressions)

1. paus (pause), d v s brist på tal och gest under en tur. Observera att endast OCM-pauser, d v s pauser med OCM-funktion ska kodas (Pauser räknas här stipulativt som en sorts uttryck.)

**Exempel:** de betyder att [//]<sub>pause</sub> alla försöker va ett steg före hela tiden

2. enkla OCM-uttryck (simple\_ocm\_expression), t ex eh, äh, m, liksom, eller, nä.

**Exempel:** ja kom å tänka på [äh]<sub>simple\_OCM\_expr</sub> torpet

3. explicita OCM-fraser (explicit\_ocm\_phrase), t.ex. vad heter det, rättare sagt, så att säga

4. andra OCM-ljud (other\_ocm\_sound) som är svåra att klassificera, t.ex. smackande, suckar mm

- Grundläggande OCM-operationer (basic\_ocm\_operations)

1. Förlängning av kontinuerliga ljud (lengthening\_of\_continuant), dvs ljud som går att hålla ut.

**Exempel:** ja hade ju hoppat över dom här konstiga figurerna fö ja [inte:]<sub>lengthening\_of\_continuant</sub> ja0 alltså

2. Självavbrott (self\_interruption), d v s talaren avbryter sig själv mitt i ett ord eller mitt i en fras. Självavbrott kan alltså förekomma antingen mitt i ett ord, vanligtvis märkt med + i transkriptionen, eller mellan två ord. Det senare fallet indikeras inte explicit i transkriptionen, utan syns vanligtvis enbart som en plötsligt avbrott i den föregående syntaktiska strukturen, eventuellt följt av enkla OCM-strukturer som t ex pauser och tvekljud. Vid kodning av självavbrott mellan två ord markeras det ord som förekommer omedelbart före avbrottet

**Exempel:** så [ma+]<sub>self\_interruption</sub> a just de a å

**Exempel:** men [vi]<sub>self\_interruption</sub> ja1 tänkte att vi får väl lösa de0

3. Självrepetition (`self_repetition`), d v s talaren upprepar något som han/hon precis har sagt. Det som upprepas kan vara antingen ett ord eller en hel fras, och eventuellt kan OCM-strukturer dyka upp mellan upprepningarna. Observera att om något annat än OCM-strukturer förekommer mellan upprepningarna, så räknas det inte som självrepetition.

Denna definition kan fångas i ett enkelt schema för självrepetition:

**Schema SR:** A (OCM) A

Detta schema tolkas så att ett ord eller en fras A följd av samma ord eller fras A, eventuellt med (grundläggande) OCM-strukturer mellan, utgör ett fall av upprepning<sup>3</sup>.

**Exempel:** de e valt bara bara för att de ska

**Exempel<sup>4</sup>:** de e valt bara för [**bara för**]<sub>self\_repetition</sub> att de ska

**Exempel<sup>5</sup>:** de e valt bara ä1 / [**bara**]<sub>self\_repetition</sub> för att de ska

Kodning av dessa uttrycksstrukturer görs enligt följande procedur:

1. Märk ut ett grundläggande OCM-uttryck eller ett uttryck (ord) som realiserar en grundläggande OCM-operation (Denna sekvens motsvaras i exemplen ovan av text i fetstil.) Observera att i fallet "självrepetition" markeras endast upprepningen/upprepningarna.
2. Välj den kod vars definition passar det märkta segmentet.

Dessa drag kan förekomma isolerade eller i kombination. När de kombineras kan de appliceras på samma segmentella uttryck, t.ex. förlängning av en vokal i ett enkelt OCM-uttryck som i sig självt uttrycker en choice-funktion

---

<sup>3</sup>Det två förekomsterna av A behöver inte vara helt identiska; de kan tex vara olika uttalsvarianter av samma ord/uttryck, eller någon förekomst kan vara avbruten. Detta gäller även de scheman som ges i kapitel 4.2.

<sup>4</sup>Detta exempel är konstruerat.

<sup>5</sup>Detta exempel är konstruerat.

(**äh** ⇒ **ä:h**). De kan också förekomma i sekvens, t.ex en paus följd av ett enkelt OCM-uttryck (**// äh**).

**Exempel:** för att inte **[ääh]**<sub>simple\_ocm\_expression</sub> **[//]**<sub>pause</sub> **[eh]**<sub>simple\_ocm\_expr</sub>  
för att hålla en del gröder vid liv

## 4.2 Komplexa OCM-operationer (complex\_ocm\_operations)

Komplexa OCM-operationer<sup>6</sup> är ett samlingsnamn på olika sätt att modifiera den språkliga strukturen. Samtliga komplexa OCM-operationer har alltså **change**-funktion, och behöver därför inte funktionskodas. Komplexa OCM-operationer involverar alltid ett självavbrott, som ofta kompletteras med ett antal andra grundläggande OCM-strukturer.

Nedan anges schematiska definitioner av de komplexa OCM-operationerna. *Dessa definitioner är inte tvingande*, d v s de behöver ej alltid följas exakt. Däremot bör kodaren alltid anteckna de fall där ett yttrandeselement kodas såsom realiserande en viss OCM-operation trots att segmentet inte uppfyller den schematiska definitionen. Helst bör hela yttrandet nedtecknas, tillsammans med den aktuella transkriptionens namn. Då kodningen görs med TRACTOR bör även de siffror som anger segmentets position i transkriptionen nedtecknas. Dessa fall bör sedan diskuteras gemensamt av kodningsgruppen då de i vissa fall kan motivera ändringar i kodningsmanualen.

Observera att (själv)avbrott i de schematiska definitionerna symboliseras med “+”. Detta innebär inte att dessa avbrott alltid måste vara indikerade med “+” i transkriptionen; ibland kan avbrottet ske efter att ett helt ord uttalats, och då syns avbrottet eventuellt bara som en paus.

I definitionerna förekommer också symbolen (OCM). Detta betyder att här kan eventuellt (grundläggande) OCM-strukturer förekomma. Generellt sett kan OCM-strukturer alltid förekomma efter självavbrott.

---

<sup>6</sup>Då det inte finns några komplexa OCM-uttryck, sammanfaller kategorierna ”komplexa OCM-uttryck” och ”komplexa OCM-operationer”.)

Förkortningarna “LC” och “RC” kan utläsas “Left Context” respektive “Right Context”. Siffrorna “1” och “2” indikerar om det rör sig om den första eller andra instansieringen av ett schematiskt element. Den andra instansieringen (t ex LC2) utgör alltså en upprepning av den första (i detta fall LC1).

Vissa schematiska element förekommer inom parenteser. Dessa parenteser indikerar att elementet är optionellt, d v s att det kan, men måste inte, förekomma. Observera dock att om den ena instansieringen av ett schematiskt element (t ex RC1) finns med så måste även den andra instansieringen (i detta fall RC2) finnas med

För att öka läsligheten i exemplen nedan har all funktionell kodning och all strukturkodning utöver den beskrivna operationen utelämnats.

1. Strykning (deletion) förekommer då material tydligt utelämnas i upprepningen.

**Schema D:** LC1 deleted (RC1)+ (OCM) LC2 (RC2)

**Exempel:** men [de e]<sub>LC1</sub> [ju]<sub>deleted</sub> [farlit]<sub>RC1</sub> // [de e]<sub>LC2</sub> [farlit]<sub>RC2</sub>  
för naturen

**Exempel:** men [de e]<sub>LC1</sub> [ju]<sub>deleted</sub> // [de e]<sub>LC2</sub> farlit för naturen<sup>7</sup>

2. Insättning (insertion) förekommer då material tydligt skjuts in (mitt) i upprepningen.

**Schema I:** (LC1) RC1+ (OCM) (LC2) inserted RC2

**Exempel:** de måste han va för att han [snabbt]<sub>LC1</sub> [på+]<sub>RC1</sub> [snabbt]<sub>LC2</sub>  
[ska kunna ta beslut]<sub>inserted</sub> [på]<sub>RC2</sub> en tiondels sekund

Insättning kan även förekomma utan återknytande<sup>8</sup>.

**Exempel:** men [vi0]<sub>RC1</sub> [ja1 tänkte att]<sub>inserted</sub> [vi0]<sub>RC2</sub> får väl lösa de<sup>9</sup>

---

<sup>7</sup>Detta exempel är konstruerat.

<sup>8</sup>När en del av det som sas före ett självavbrott uprepas efter avbrottet talar man om *återknytande* (eng. resumption).

<sup>9</sup>Notera att detta också kan tänkas vara ett fall av substitution snarare än insättning, där ja1 ersätter vi0. Vilken operation det är frågan om måste kodaren själv avgöra genom att använda sina språkliga intuitioner. I sådana fall är det ofta till stor hjälp att lyssna på ljudinspelningen.

3. Utbyte (substitution) förekommer då material tydligt byts ut i upprepningen.

**Schema S** (LC1) substituted (RC1)+ (OCM) (LC2) substitute (RC2)

**Exempel:** så de [känns]<sub>LC1</sub> [som]<sub>substituted</sub> [de e va+]<sub>RC1</sub> [känns]<sub>LC2</sub> [att]<sub>substitute</sub> [de e valt]<sub>RC2</sub> bara för att

Även substitution kan förekomma utan återknytning, om det ersatta elementet (substituted) och det ersättande elementet (substitute) har samma roll i satsen.

**Exempel:** de0 [blir]<sub>substituted</sub> [väldit]<sub>RC1</sub> [låter]<sub>substitute</sub> [väldit]<sub>RC2</sub> jobbit

Ibland förekommer substitution med enklast tänkbara struktur, där även elementet RC i schemat ovan är utelämnat. Även här måste det ersatta elementet och det ersättande elementet ha samma roll i satsen.

**Exempel:** ska vi återkalla lite grann [om]<sub>substituted</sub> / [AV]<sub>substitute</sub> de vi talade om förra veckan

**Exempel:** han talade om [våra]<sub>substituted</sub> el eller [vår]<sub>substitute</sub> nya datoriserade värld

4. Omkastning (reordering) förekommer då material tydligt kastas om i upprepningen.

**Schema R:** LC1 re\_a re\_b RC1+ (OCM) LC2 re\_b re\_a RC2

**Exempel:** men sen [hade]<sub>LC1</sub> [ja]<sub>re\_a</sub> [inte]<sub>re\_b</sub> [lä+]<sub>RC1</sub> [hade]<sub>LC2</sub> [inte]<sub>re\_b</sub> [ja]<sub>re\_a</sub> [läst]<sub>RC2</sub> dom siderna

Kodning av dessa operationer görs enligt följande procedur:

Givet att du hittat ett yttrandesegment som uppfyller någon av de ovanstående definitionerna, gör följande:

1. Markera relevant del del (ord eller fras) av segmentet
2. Välj den kod som bestäms av typen av struktur och segmentets schematiska del i definitionen av strukturen, t ex *inserted* eller *RC1*.

## References

- [1] Larsson, S. (1997): TRACTOR användarmanual. Dept. of linguistics, Göteborg University.
- [2] Larsson, S. & Sofkova, S. (eds) (1996): Modifierad standardortografi version 4. Dept. of linguistics, Göteborg University.
- [3] Nivre, J. & Sofkova, S. (eds) (1996): Transcription Standards version 4. Dept. of linguistics, Göteborg University.