# Talen en Automaten Case Study: Objekts R Kool

Rob Föllings & Jip J. Dekker

20 Januari 2012

## 1 Beschrijving

Objekts R Kool, oftewel ORK, is een object georiënteerde programmeer taal. De taal is in 2005 ontworpen door Gregor Richards. ORK is een hele verbale taal, gebaseerd op de engelse spreektaal en vertelt in het engels dan ook precies wat er gebeurd. Dit maakt het voor programmeurs erg grappig en voor anderen redelijk begrijpelijk. ORK maakt uitsluitend gebruik van objecten met de bijbehorende functies en twee typen variabelen:

1. number - Dit vertaalt zich naar een getal.
2. word/phrase/sentence - Alle drie vertalen zich naar een string.

In de taal zijn een paar standaard classen en bijbehorende functies opgenomen die in- en uitvoer, en vergelijken mogelijk maken. Doordat ORK zo object georiënteerd is, is er een hoop code nodig om zelfs de simpelste programmaatjes te schrijven. Een paar voorbeelden hiervan. Het voor de programmeurs onder ons bekende "Hello World!"programma, dat enkel een datzelfde zinnetje output, ziet er als volgt uit:

```
When this program starts:
There is a scribe called Writer.
Writer is to write "Hello, world!"
```

Dit is een erg simpel programma, maar zoals te zien hebben we hier al drie regels voor nodig. Hieronder is een moeilijker voorbeeld te zien, dit programma heeft 2 classen, zodra dit programma start wordt de gebruiker om een "Squeaky sound"gevraagd, waarna een muis genaamd Gerald wordt opgegeten door een kat genaamd "Lucifer Sam"waarbij de muis deze "Squeaky sound"maakt:

```
There is such a thing as a mouse.
A mouse can be_eaten.
A mouse has a status which is a word.
A mouse has a voice which is a scribe.
A mouse has an input which is an inputter.

When a mouse is to be_eaten:
There is a word called squeaky sound.
input is to readOne squeaky sound.
If input says it's done then status is "eaten".
voice is to write squeaky sound.

There is such a thing as a cat.
A cat can eat a mouse.
A cat has a Lingo which is a linguist
```

```
When a cat is to eat a mouse:
The mouse is to be_eaten.
Lingo's first operand is the mouse's status.
Lingo's second operand is "eaten".
Lingo is to compare.
If Lingo says it's not equal then I am to loop.

When this program starts:
There is a cat called Lucifer Sam.
There is a mouse called Gerald.
Gerald's status is "alive".
Lucifer Sam is to eat Gerald.
```

Zoals te zien, zit dit programma al op 5 paragrafen terwijl dit een erg simpel programma is.
Wil je nog meer weten over ORK en de standaard classen die beschikbaar zijn, bekijk dan de engelse ORK leesmij:
ORK README

# 2 Regulier?

ORK is geen reguliere taal.

*Bewijs.* Stel dat ORK, hierna geschreven als taal $L$, wel een reguliere taal is. Dan is er een niet-deterministische automaat met $k$ toestanden die taal $L$ accepteert. In ORK moet een functie één keer gedeclareert worden voordat deze kan worden gebruikt. Het declareren van functie i schrijf ik als $a_i$ en het gebruik als $b_i$. Dus als $w \equiv a_1a_2...a_ka_{k+1}b_1b_2...b_kb_{k+1}$ dan $w \in L$, omdat $L$ een reguliere taal is en $|w| > k$ zegt het pomplemma dat met $w = uvw$ en $|uv| < k$ dat $v$ herhaalt mag worden zonder buiten de taal te treden. Doordat $uv$ binnen de eerste $k$ moeten vallen weten we dat $v$ bestaat uit één of meerdere declareringen. Maar ik mag een functie slecht eenmaal declareren, dus de gepompte woorden vallen buiten taal $L$. Dit is in tegenspraak met het pomplemma, dus taal $L$ is niet regulier. □

# 3 Contextvrij?

ORK is een contextvrije taal.

*Bewijs.* Een taal is contextvrij als deze beschreven kan worden door een contextvrij grammatica.
Accolades geven zelf te bepalen namen aan.
Rechte haken geven aan dat het optioneel is.

**S** → **C** "When this program starts:"**F**
**F** → "{variable} is {variable value}."**F** | "I have a[n] {object} called {object name}."**OF F** | "There is a[n] {object} called {object name}."**OF F** | $\lambda$
**OF** → "{object} is to {function} [{parameter}]."**OF** | $\lambda$
**C** → **C** "There is such a thing as a[n] {classname}"**CF** | $\lambda$
**CF** → "A {classname} can {function} [a[n] {parameter type}]."**CF FV** | **CF** "A class has [a[n]] {variable name} which is a {variable type}" | $\lambda$
**FV** → "When a class is to {function} [a[n] {variable type}]:"**F**

Iedere Terminal heeft hierin de volgende betekenis:
**S:** Start

**F:** Functie
**OF:** Object Functie (Een object is onderdeel van een bepaalde class)
**C:** Class
**CF:** Class Functie
**FV:** Functie Voorschrift

Naast deze grammatica bevat ORK nog wat standaard classes en bijbehorende functies, met deze standaard classes en deze grammatica zijn alle programma's te maken (Alhoewel de beschikbare loops je een hoop tijd en herhalende code zou bespraren). Dus is bewezen dat ORK een contextvrije taal is. □