Informatie delen

|  |  |  |
| --- | --- | --- |
| Joop Ringelberg | 05-03-18 | Versie: 1 |

# Introductie

Als een nieuwe gebruiker bij een context betrokken wordt, moet hij voorzien worden van de juiste informatie: de context, sommige rollen, sommige properties daarvan, en misschien nog wel meer.

De vraag is: welke informatie moet worden verstuurd? Elke Perspectives client moet deze vraag beantwoorden. Telkens als iemand bij een context wordt betrokken, moeten àlle andere betrokkenen die vraag beantwoorden. Elk kan immers als enige over informatie beschikken die de nieuwe betrokkene nodig heeft.

Een simpele strategie zou zijn: verstuur alles wat je hebt. Maar dat conflicteert met een non-functional van Perspectives, namelijk dat elke betrokkene slechts beschikt over informatie waar hij krachtens het model iets mee kan doen (waarvoor hij een actie heeft die die informatie in het lijdend- of meewerkend voorwerp gebruikt).

# Acties en queries

De nieuwe betrokkene – laten we hem B noemen - speelt een rol in de context. Bij deze rol hoort een aantal acties, in die zin dat de betreffende rol het onderwerp ervan vult. B moet in elk geval beschikken over de lijdende voorwerpen van al die acties. Lijdend voorwerpen zijn rollen van de context. Maar het kan om een berekende rol gaan en dan moet B die berekening zelf kunnen uitvoeren.

Een berekening is een query. De client moet dus de query analyseren om vast te stellen welke contexten en rollen in de uitvoering ervan ‘bezocht worden’ (d.w.z. dat ze in een tussenresultaat van de query voorkomen).

In grote lijnen geeft dit de richting aan waarmee we de vraag: wat moeten we versturen? zullen beantwoorden. Maar er is meer. Eerst een voorbeeld.

# Aangiften met verdachten

Als query nemen we: alle aangiften waarbij medewerker X de Verbalisant is en er een Verdachte is. Laten we als context een intern onderzoek naar X veronderstellen, waar rechercheur B bij betrokken wordt. Welke informatie ontvangt B?[[1]](#footnote-1)

Een client kan eenvoudig een lijst relevante aangiften samenstellen, door zelf de query uit te voeren. Kan hij ermee volstaan de queryresultaten naar B te sturen? Laten we een paar gevallen bekijken.

1. Er is (op de client) een Aangifte A1 waarvan de client niet weet wie de Verbalisant is. Dat hoeft niet te betekenen dat er geen Verbalisant voor A1 (bekend) is. Een andere client kan weten wie de Verbalisant is. Maar, misschien weet díe client niet wie de Aangever is! Kortom, om aan de veilige kant te blijven – om te voorkomen dat B straks een onvolledig resultaat berekent! – moeten we A1 toch óók naar B sturen.
2. Er is een Aangifte A2 waarvan de client niet weet wie de Aangever is. Met dezelfde redenering als boven moeten we concluderen dat we óók A2 naar B moeten sturen.

Kortom, B kan vanuit diverse andere betrokkenen bij het onderzoek naar X aangiften ontvangen waaraan een Verbalisant of Verdachte ontbreekt. Laten we aannemen dat B voldoende informatie ontvangt om uiteindelijk van elke aangifte de Verbalisant te kennen. Een aantal van hen zal blijken niet door X te zijn geverbaliseerd en zijn dus irrelevant voor het onderzoek.

Een aantal zal blijken nog steeds geen Verdachte te hebben. Die aangiften komen niet in het query resultaat en zijn dus eveneens irrelevant voor het onderzoek.

Het is, gegeven deze situaties, dus onvermijdelijk dat B óók irrelevante informatie ontvangt. Maar het kan beter.

# Volledige of onvolledige informatie

Als een betrokkene een actie heeft met een bepaalde rol als lijdend voorwerp, zal hij van andere betrokkenen elke instantie van die rol toegestuurd krijgen. Hij heeft *volledige informatie* over die rol.

Maar als het lijdend voorwerp een berekende rol is, waarbij de berekening een filter is over een rol van de context, krijgt hij géén volledige informatie. Immers, anderen zullen hem uitgefilterde instanties (degenen die niet voldoen aan het filtercriterium) niet toesturen.

Uit het model blijkt dus op eenvoudige wijze of een betrokkene volledige of onvolledige informatie heeft over een rol[[2]](#footnote-2).

Dit is een vorm van *auto-epistemische* kennis. De client moet die kennis gebruiken als hij voor een nieuw betrokken gebruiker bepaalt welke contexten, rollen en properties hij hem gaat toesturen.

# Open en gesloten

Veronderstel dat iemand zijn gezin beschrijft. Hij geeft aan dat hij twee zonen heeft en sluit af. Is het mogelijk dat deze persoon ook nog dochters heeft? Of nog meer zonen? Jawel, maar het is onwaarschijnlijk. Des te meer als we hem vragen volledig te zijn. Voor praktische doeleinden kunnen we aannemen dat iemand zijn eigen autoriteit op dit gebied is.

We kunnen dit veralgemeniseren naar situaties waarin een informant weet dat bepaalde informatie volledig is, of, omgekeerd, weet dat hij dat niet weet. Dit soort inzicht kunnen we scharen onder auto-epistemisch redeneren.

Omdat de gegevens die we met Perspectives bijhouden zo simpel zijn gestructureerd, kunnen we overzichtelijk alle gevallen nagaan met betrekking tot deze auto-epistemologische aspecten.

Eerst beschouwen we een voorbeeld, namelijk (biologisch) moederschap. Zodra we weten wie iemands biogische moeder is, weten we ook dat we alles daarover weten. Immers, iedereen heeft slechts één biologische moeder (moderne vruchtbaarheidstechnologie daargelaten). Dat onderwerp is *gesloten*.

We noemen een rol van een context *gesloten* als we weten dat er geen instanties van een rol meer bij zullen komen[[3]](#footnote-3). In Perspectives kunnen we de cardinaliteit van een rol aangeven als enkel- of meervoudig. Zoals het moederschap-voorbeeld aangeeft, zodra we een instantie hebben van een rol met enkelvoudige cardinaliteit is de rol in die context gesloten.

Maar in het voorbeeld van de vader die zijn gezin beschrijft, zagen we dat ook een rol met meervoudige cardinaliteit best gesloten kan zijn – op autoriteit van de informant. Die auto-epistemische kwaliteit moeten we expliciet vastleggen om hem te kunnen gebruiken. Het valt immers niet af te leiden.

## Open en gesloten contexten, rollen en properties

Een rol is dus per definitie gesloten in een context als we alle mogelijke instanties ervan kennen. Hetzelfde geldt voor een property waarvan we alle mogelijke waarden kennen. In beide gevallen moeten we onderscheid maken tussen rollen (properties) met enkel- en meervoudige cardinaliteit.

Gewapend met deze uitgangspunten kunnen we definiëren wat het betekent dat een rol-instantie gesloten is: dan zijn al zijn properties gesloten èn is zijn binding bekend.

De definitie van een gesloten context volgt onmiddellijk: dat is een context waarvan alle rollen gesloten zijn.

Merk op dat een gesloten context niet alleen maar gesloten rol-instanties heeft! Het gesloten zijn van een rol en een rol-instantie zijn verschillende dingen.

## Gesloten is een relatief begrip

Bij het voorbeeld van het interne politie-onderzoek gaat het maar om twee rollen: Verbalisant en Verdachte. Het kan ons niets schelen of en hoeveel Aangevers er zijn, enzovoort. Dat is zo omdat de query waar we vanuit gingen, alléén gebruik maakt van die twee rollen. We noemen daarom *een context gesloten ten opzichte van een query* als alle in de query voorkomende rollen gesloten zijn.

Op dezelfde manier kunnen we *een rol-instantie gesloten verklaren ten opzichte van een query*.

## Open en gesloten is niet voor iedereen hetzelfde

Een enkelvoudige rol waarvan een instantie beschikbaar is, is gesloten. Niet elke betrokkene hoeft over die instantie te beschikken. Maar elke betrokkene kan (in het model) nagaan of hij over een instantie zou móeten beschikken. Zo ja, dan kan hij uitsluitsel geven over het open of gesloten zijn van die rol. Zo nee, dan kan hij dat niet.

Hieruit blijkt dat verschillende betrokkenen anders kunnen oordelen over het open of gesloten zijn van dezelfde rol (en mutatis mutandis, van een property).

Bij een rol met meervoudige cardinaliteit registreren we expliciet of hij open of gesloten is. Een betrokkene die over zo’n rol moet beschikken, kan die registratie aflezen.

Stel dat een betrokkene via een query beschikt over instanties van een rol die aan voorwaarden voldoen. Vanwege de voorwaarde hoeft deze betrokkene dus niet over alle instanties van die rol te beschikken. Desondanks is ook voor deze betrokkene de rol gesloten. Er komen immers geen instanties meer bij.

# Aangiften met verdachten – open of gesloten?

We hebben een voorbeeld besproken waarbij we werkten met een query naar alle aangiften waarvan een medewerker X de Verbalisant is en er een Verdachte is. We bespraken het voorbeeld van een Aangifte A2 waarvan de client niet weet wie de Aangever is. Zo’n aangifte moet toch naar de nieuwe betrokkene gestuurd worden.

Maar dat hoeft niet, als A2 gesloten is ten opzichte van de query. Concreet: als de client weet dat de rol Aangever gesloten is, met andere woorden, hij weet dat geen enkele andere client een Aangever bij die aangifte heeft. A2 hoort definitief niet thuis in het queryresultaat. We hoeven hem niet te versturen.

Hetzelfde geldt voor het andere voorbeeld, A1 (maar het is onwaarschijnlijk dat er een aangifte zou zijn zonder Verbalisant).

Kortom: naarmate we meer weten over open- of geslotenheid van contexten en rollen ten opzichte van de queries die een nieuwe betrokkene moet kunnen uitvoeren, kunnen we zuiniger zijn met het oversturen van informatie, zonder dat de mogelijkheden van de nieuwe betrokkene op enigerlei wijze gecompromitteerd worden.

# Volledige queries

Stel je een query Q voor die bestaat uit één stap, namelijk het opvragen van een rol R bij een context C. En stel dat we weten dat C gesloten is ten opzichte van Q – oftewel, we weten dat we elke instantie van R bij C kennen. In zo’n geval noemen we *het queryresultaat van Q volledig*.

Maar als de query uit twee stappen bestaat en beide stappen zijn gesloten ten opzichte van Q, dan is zo’n query ook volledig.

Het blauw weergegeven tussenresultaat brengt geen elementen in het eindresultaat voort en hoeft daarom niet te worden opgestuurd.

Kortom, als alle stappen gesloten zijn ten opzichte van de query, is de query volledig. We kunnen dan volstaan met het query-resultaat en alle tussenresultaten die nodig zijn om het eindresultaat te berekenen. Dat kan aanzienlijk minder zijn dan àlle tussenresultaten. Denk bijvoorbeeld aan een query die alle voltooide aangiften oplevert. Als we weten dat de property ‘voltooid’ gesloten is[[4]](#footnote-4), hoeven we alleen maar voltooide aangiften te versturen.

# Onvolledige queries

Wat is de consequentie van een query met een stap die nog open is? In de figuur is te zien hoe het blauwe tussenresultaat geen elementen van het eindresultaat voortbrengt. Maar als de bijbehorende stap nog open is, kan het zijn dat een andere client over informatie beschikt waardoor wèl een element aan het eindresultaat wordt toegevoegd. Daarom moet dit tussenresultaat toch worden opgestuurd.

1. We kunnen ook de vraag stellen: van wíe ontvangt B informatie? Die zal moeten komen van andere betrokkenen bij het onderzoek. Daar zullen mensen bij moeten zitten die op de één of andere manier betrokken zijn bij Aangiften waarbij X de verbalisant is. [↑](#footnote-ref-1)
2. Hetzelfde kan gelden voor een property, maar is minder waarschijnlijk. [↑](#footnote-ref-2)
3. Om een fijn punt te maken: dat hoeft niet te betekenen dat elke betrokkene over al die instanties beschikt! [↑](#footnote-ref-3)
4. Deze functionele property zou een boolean waarde hebben. We mogen zo’n property alleen gesloten verklaren, als we altijd een waarde invullen! Met andere woorden, we mogen hier niet de Closed World Assumption toepassen door ervan uit te gaan dat géén waarde voor ‘voltooid’ betekent “niet voltooid”. [↑](#footnote-ref-4)