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.

Als een client onvolledige informatie heeft over een rol die in het criterium van een queryfilter gebruikt wordt, heeft dat gevolgen voor wat hij wel of niet verstuurt. Laten we de aangifte A2 uit het voorbeeld hierboven nemen, waarvoor geldt dat de client niet weet wie de Verbalisant is. Als hij volledige informatie van die rol heeft, hoeft hij A2 niet aan B te versturen. Immers, X is niet de Verbalisant, want dan had de client het geweten. Daarentegen, als hij ònvolledige kennis heeft van de Verbalisant rol, stuurt hij A2 wel aan B. Een andere betrokkene kan misschien wel een invulling van de Verbalisant rol hebben (maar misschien weer niet van de Aangever rol).

We zeggen dat de waarheidswaarde van het criterium *zeker* is als de informatie van de erin gebruikte rol volledig is; anders is de waarde *onzeker*. In die termen: als een entiteit onzeker wordt uitgefilterd, stuurt de client het toch op.

## Filter criteria bestaand uit logische expressies

Als een filter criterium bestaat uit de conjunctie van twee criteria, is de zekerheid van het criterium de conjunctie van de zekerheid van de criteria. Als we de query bijvoorbeeld uitbreiden tot: alle aangiftes waarbij X de Verbalisant is en die zowel een Aangever als een Verdachte hebben, dan is er sprake van een conjunctie van criteria. Nu geldt dat als de client een aangifte wegfiltert, hij hem tòch verstuurt als hij over één rol onvolledige kennis heeft. Zijn conclusie is dan immers niet zeker.

Bij een disjunctie ligt het net iets ingewikkelder. Daar geldt dat de disjunctie zeker is in twee onderscheiden gevallen:

* als minstens één disjunct wáár èn zeker is (de disjunctie is dan ook waar en zeker);
* of als beide disjuncten onwaar en zeker zijn (de disjunctie is dan ook onwaar en zeker).

Bij een ontkenning draait de waarheidswaarde natuurlijk om, maar de zekerheid ervan blijft hetzelfde.

# 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 de client volledige kennis heeft van R - 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 zulke stappen bestaat en de client heeft volledige kennis over beide rollen, 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 bij elke stap de kennis volledig is, 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’ volledig is[[3]](#footnote-3), hoeven we alleen maar voltooide aangiften te versturen.

## Terugredeneren

In de bovenstaande figuur zien we hoe dit proces over drie stappen kan verlopen. Merk op dat in de tweede kolom van links een blauwe cirkel voorkomt. Deze cirkel is blauw omdat de entiteit die hij representeert geen resultaat voortbrengt in de rechterkolom (het eindresultaat van de query) onder de betreffende querystap.

Maar in de linker kolom zien we een groene cirkel staan. Die staat voor een entiteit die wèl een resultaat voorbrengt in de kolom ernaast, maar toch weggelaten kan worden. Er is namelijk geen pad vanuit die entiteit naar een eindresultaat.

Een algoritme om precies te bepalen welke tussenresultaten opgestuurd moeten worden. We noemen het tussenresultaat van de eerste stap T1 en de eerste stap zelf S1.

1. Begin bij het eindresultaat Tn en noem dat R.
2. Pas Sn omgekeerd toe op R en noem het resultaat R’
3. Neem de doorsnede van R’ en Tn-1 en noem dat R. R moet worden opgestuurd.
4. Herhaal het algoritme vanaf stap 2, waarbij n = n-1.

# Onvolledige queries

Wat is de consequentie van een query met een rol waarover de client slechts onvolledige kennis heeft? In de figuur is te zien hoe het blauwe tussenresultaat geen elementen van het eindresultaat voortbrengt. Maar als de bijbehorende rol onvolledig is voor de client, 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. Deze functionele property zou een boolean waarde hebben. Kennis kan volledig zijn, terwijl we ook de Closed World Assumption toepassen. Beide zijn onafhankelijk. In dat geval geldt: als de property geen waarde heeft, nemen we aan dat de waarde false is. [↑](#footnote-ref-3)