Versie 1

Dit project situeert hem omtrent grid applicaties. Dit zijn applicaties die uitgevoerd worden op een gedistribueerd netwerk waarbij de knopen van dit netwerk elk een individueel doel hebben en onderling communiceren om een collectief doel te bereiken. Deze applicaties zijn vaak kritisch van aard. Een voorbeeld zijn verkeerslichten die onderling communiceren.

Vandaag is er geen technologie voorhanden die ons toestaat om deze aplicaties efficient te schrijven.

In dit project zullen wij een domeinspecifieke taal ontwikkelen die de programmeur toestaat om 1) efficient met het netwerk om te psringen 2) statische verificatietechnieken voor de communicatie code te controleren als laatste 3) een runtime die toelaat om code consistent en real-time up te daten.

Comments

- Minder introductie spenderen aan grid applicaties
- begin met een voorbeeld
- Enkel problemen goed uitleggen en oplossing in 1 zin zeggen : we gon' fix this

Versie 2

Dankzij mijn projectvoorstel zal het makkelijker maken om grid applicaties te programmeren. Een voorbeeld van een grid applicatie is het verkeerslichten netwerk. De verkeerslichten hebben een invidiuele taak, namelijk het bepalen wanneer groen of rood licht geven aan de hand van ingebouwde sensors. Onderling communiceren de verkeerslichten om op een intelligente manier een optimale doorstroom te verzekeren. Deze applicatie heeft uiteraard een nultolerantie voor fouten in de en communicatie.

Als we zo een applicatie vandaag willen programmeren merken we echter op dat er geen geschikte technologie voorhanden is om dit efficient te doen.

Het updaten van de software op verkeerslichten kan nu enkel via periodieke updates van specifieke regels. Dit willen wij consistent en real-time mogelijk zodat nieuwe software propageert ovet het netwerk.

De correctheid van communicatie tussen de verkeerslichten moet manueel gecontroleerd worden door repetitieve code. Dit willen wij aanpakken door verificatietechnieken los te laten op de programmatekst alvorens het programma uitgerold wordt.

En als laatste kan het netwerk niet op een efficiente en intelligente manier aangesproken worden vanuit een programmatekst. Dit willen wij doen door taalconstructies te ontwikkelen die dit wel mogelijk maken.

Comments

- Less focus on traffic lights
- Netwerk topologie first class maken (Is dit duidelijk genoeg?) Voorzie hier een vraag op!
- IP addressen gebruiken maakt het probleem duidelijk maken.
- Intelligente manier verduidelijken met een voorbeeldje
- Probeer er voor te zorgen dat ze een idee hebben van hoe code er zou uitzien
- Eerste zin minder arrogant maken.
- Te lange zinnen vermijden.
- Zullen en willen vermijden. Imperatief zeggen.
- Logische eenheden selecteren etc

Versie 3

In dit projectvoorstel zal ik het het makkelijker maken om grid applicaties te programmeren. Een voorbeeld van een grid applicatie is het verkeerslichten netwerk. De verkeerslichten hebben een invidiuele taak, namelijk het bepalen wanneer groen of rood licht geven aan de hand van ingebouwde sensors. Maar ook onderling communiceren de verkeerslichten om op een intelligente manier een optimale doorstroom te verzekeren. Deze applicaties hebben uiteraard een nultolerantie voor fouten in de en communicatie. De communicatiepatronen moeten door de programmeur correct uitgedrukt worden.

Zo een soort van applicaties vandaag schrijven is ingewikkeld en omvat veel accidentele complexiteit. Wij pakken deze aan op 3 fronten.

1. Het updaten van software op de nodes van een gridapplicatie zullen we realtime en consistent mogelijk maken. De nodes zullen dan zelf de nieuwe software propageren.
2. De correctheid van de uitgedrukte communicatiepatronen moet statisch gecontroleerd worden. Dit zorgt er voor dat de communicatiepatronen dezelfde verificatie genieten als de applicatie code (static typing)
3. Als laatste willen wij het netwerk een first-class citizen maken in de applicatielogica. Dit zorgt er voor dat de programmeur op een eenvoudige manier het netwerk kan aanspreken in logische eenheden. Op die manier hoeft niet omgesprongen te worden met IP addressen en dergelijke.

Comments

- Het spreekt niet. Minder schrijftaal! Vermijd zinnen die niets zeggen. Begin direct over voorstel. (Geen "goeie morgen.."). Ze weten waar mijn projectvoorstel over gaat. Bijv. beginnen met "Grid applicaties zijn.."

- "In dit projectvoorstel er uit laten"
- Meer eenvoudige zinnen (spreektaal)
- Het tweede deel is niet meer zo duidelijk. Probeer toch nog met verkeerslichten te werken "Bijvoorbeeld alle verkeerslichten die naburig op groen staan"
- Minder in detail treden over wat ik in detail ga doen in mijn oplossing.
- Probeer structuur te bewaren maar probeer het makkelijker te formuleren.
- Voorbeeldje ipv "logische eenheden" → "Nodes selecteren op basis van logische queries"

Versie 4

Grid applicaties zijn programma's die uitgevoerd worden op een netwerk en hebben enkele specifieke eigenschappen. Een voorbeeld van een grid applicatie is het verkeerslichten netwerk. De verkeerslichten hebben een invidiuele taak, zijnde groen of rood licht geven op basis van interactie met voetgangers en/of sensoren. En onder elkaar communiceren ze om het verkeer soepelder te laten doorstromen. Een belangrijke eigenschap van deze software is dat er zeker geen fouten mogen optreden in de communicatie onderling en in de applicatielogica.

Als je de dag van vandaag een grid applicatie wil schrijven merk je al gauw dat de huidige programmeertalen tekort schieten op verschillende vlakken. We gaan deze dus aanpakken en samenbrengen in een enkel raamwerk.

- Een knoop in het netwerk moet "live" kunnen geupdated worden op een consistente manier. Bijv. het updaten van de software van verkeerslichten. Het is ondenkbaar dat we "eventjes" het gehele netwerk offline halen en dan alle lichten elk om beurt gaan fysiek updaten.
- De programmeur zou enkel moeten rekening houden met de applicatie. Allerhande foutopvangende code of repetitieve code draagt dus niets bij aan de betekenis van de applicatie en moet vermeden kunnen worden.
- Als laatste moet het natuurlijk mogelijk zijn om eenvoudig met het netwerk om te kunnen van in het programma. We willen geen IP adressen aanspreken of complexe code schrijven om, in context van het voorbeeld, alle naburige verkeerslichten te vinden die nu op groen staan.

Comments

- Essentiele complexiteit is niet de juiste term..
- In het eerste deel bespreek je niet genoeg wat de problemen zijn van het progarmmeren van grid applicaties
- Stress minder dat er accidentele complexiteit is!
- Huidige talen zijn gewoon niet genoeg..
- De focus moet echt liggen op het probleem!

- Nog altijd te lang
- Teveel uitleg aan verkeerslichten
- Nummer problemen en zeg dit ook.
- Probeer niet te definieren wat een een grid applicatie is.
- Definieer het globaal en zeg dan "zoals bv. het verkeerslichten netwerk"
- Introduceer pas het raamwerk na de oplijsting van problemen.
- Ik-vorm gebruiken bij oplijsting van problemen
- Deze → Die
- Misschien toch "first-class citizen" gebruiken in laatste punt.

Versie 5

Grid applicaties zijn een vorm van gedistribueerd programmeren. We hebben een applicatie die runt op een licht dynamisch netwerk. De system moeten statisch geverifieerd worden om fouten tegen te gaan en als laatste is het belangrijk dat de software van elke knoop kan geupdatet worden at runtime zonder het systeem offline te halen. Voorbeelden van zo een applicaties zijn het verkeerslichten netwerk of automatische transport robots in een industriele hangar.

Als je de dag van vandaag een grid applicatie wil schrijven merk je al gauw dat de huidige programmeertalen tekort schieten op verschillende vlakken. We gaan hieruit inspiratie putten en een raamwerk ontwikkelen dat zich zal focussen op 3 pijnpunten.

- Ten eerste, moet het mogelijk zijn om een knoop in het netwerk "live" te kunnen updaten op een consistente manier. Bv. het updaten van de software van verkeerslichten. Het is ondenkbaar dat we "eventjes" het gehele netwerk offline halen en dan alle lichten elk om beurt gaan fysiek updaten.
- Ten tweede wil ik kunnen in staat zijn om mij te concentreren op de applicatie logica. Ik wil geen repetitieve foutopvangende code schrijven die er voor zorgt dat mijn mooi algoritme verstopt zit onder heel wat lijnen code.
- En als laatste wil ik het gehele netwerk als een first-class citizen aanspreken. Ik wil niet langer met IP-adressen werken of referenties moeten bijhouden naar nodes om snel te kunnen communiceren. Ik wil bijvoorbeeld alle verkeerslichten rond brussel noord aanspreken aan de hand van logische queries.

Versie 6

In mijn projectvoorstel ga ik het gemakkelijker maken om grid applicaties te programmeren. Deze zijn een subset van gedistribueerde systemen. Voorbeelden van grid applicaties zijn het verkeerslichten-netwerk en robots in een industriele hangar. Wat ga ik daar concreet aan doen?

Eerst en vooral wil ik dat het mogelijk wordt om statisch zoveel mogelijk fouten op te sporen. Dat kan

uiteraard gebeuren aan de hand van statische verificatie. Bij grid applicaties is er heel veel communicatie. Daarom wil ik dat ook mijn communicatie statisch geverifieerd wordt.

Ten tweede, al deze communicatie wil ik eenvoudig uitdrukken in mijn programma. Daarom wens ik dus taal abstracties die van het netwerk een first-class citizen maken.

En als laatste wil ik dat het ook mogelijk is om real-time consistent nodes te updaten. Ik wil bijvoorbeeld niet heel mijn netwerk offline halen om een nieuwe software versie te deployen, ik wil dat dit realtime gebeurd.