Une approche événementielle pour le développement de services multi-métiers dédiés à lassistance domiciliaire

Adrien Carteron

http://phoenix.inria.fr

Directeur de thèse : Charles Consel

Co-Directeur de thèse : Nic Volanschi

5 avril 2019

Inria
INVENTEURS DU MONDE NUMÉRIQUE

université de BORDEAUX

Introduction

Contexte

Toute information utilisée pour caractériser une entité.

Dimensions de contexte

Monde physique (état matériel).
Monde logiciel (agenda).
L'utilisateur (mesures physiologiques).
Activités de l'utilisateur (ouverture de porte).

Services **sensibles au contexte**

Utiliser le contexte.
Interagir avec l'utilisateur.

Sensibilité au contexte pour supporter l'autonomie domiciliaire

Services sensibles au contexte définis par des **experts du domaine** :

Sensibilité au contexte pour supporter l'autonomie domiciliaire

Services sensibles au contexte définis par des **experts du domaine** :

Préparer le déjeuner

Allumer le four.
Ouvrir le frigidaire.

Sensibilité au contexte pour supporter l'autonomie domiciliaire

Services sensibles au contexte définis par des **experts du domaine** :

Préparer le déjeuner

Allumer le four.
Ouvrir le frigidaire.

Lever

Présence dans la chambre.
Présence dans la cuisine.

Sensibilité au contexte pour supporter l'autonomie domiciliaire

Services sensibles au contexte définis par des **experts du domaine** :

Préparer le déjeuner

Allumer le four.
Ouvrir le frigidaire.

Lever

Présence dans la chambre.
Présence dans la cuisine.

Porte entrée ouverte non surveillée

Ouvrir la porte d'entrée.
Absence dans l'entrée.

Défis de l'autonomie domiciliaire

Fiabilité de la détection du contexte.

Multiples axes de variation (Utilisateurs, domiciles, intervenants).

Dynamicité.

Défis de l'autonomie domiciliaire

Fiabilité de la détection du contexte.

Multiples axes de variation (Utilisateurs, domiciles, intervenants).

Dynamicité.

Défis de l'autonomie domiciliaire

Fiabilité de la détection du contexte.
Multiples axes de variation (Utilisateurs, domiciles, intervenants).
Dynamicité.

Défis de l'autonomie domiciliaire

Fiabilité de la détection du contexte.

Multiples axes de variation (Utilisateurs, domiciles, intervenants).

Dynamicité.

Défis de l'autonomie domiciliaire

Fiabilité de la détection du contexte.
Multiples axes de variation (Utilisateurs, domiciles, intervenants).
Dynamicité.

Problématique

Comment couvrir les besoins des services sensibles au contexte ?

État de l'art

Fiabilité de la sensibilité au contexte :

Simulation de contexte [BJC09].

- Limité aux couches applicatives.

Placement de capteurs [Hon+13] ; [Mur+13a] ; [BCL04].

- Intrusifs (caméra, capteurs portés).
- Usage en laboratoire (Gants RFID).

État de l'art

Intervenants :

Programmation par les utilisateurs [Res+09] ; [CC16] ; [CCT11].

- Expressivité limitée.
- Généralistes.

Services d'assistance domiciliaire [Hoq+15] ; [LD15a] ; [Bae+14].

- Approches en silo.

Réactivité :

Approches de traitement événementiel [CM12].

- Complexe à mettre en œuvre.

Thèse : Unifier l'expression des contextes.

Modèle d'infrastructure

→ Fiabilité.

Thèse : Unifier l'expression des contextes.

Modèle d'infrastructure

→ Fiabilité.

Langage dédié.

→ Axes de variation.

Thèse : Unifier l'expression des contextes.

Modèle d'infrastructure

→ Fiabilité.

Langage dédié.

→ Axes de variation.

Paradigme événementiel.

→ Dynamicité.

Sommaire

Analyse de domaine

Expérimentation écologique à large échelle

Domassist

Déploiement :

- 12 mois.
- 129 personnes vivant seules.
- 82 ans en moyenne.

Expérimentation écologique à large échelle

Domassist

Déploiement :

- 12 mois.
- 129 personnes vivant seules.
- 82 ans en moyenne.

Différents services déployés (assistance, maintenance, ...).

Contextes dans l'assistance domiciliaire

Nom	Description
Alerte porte d'entrée	Entrance door is open and is unattended for 5 minutes
Réchauffer un plat surgelé	Freezer gets opened and stove gets turned on within 10 minutes or Freezer gets opened during stove is on , during lunch time
Dépendance de présence	The cupboard gets opened in the kitchen, while a presence in the kitchen is false
Échec de communication	A sensor fails to communicate for 24 hours

Commonalités :

Variabilités :

Contextes dans l'assistance domiciliaire

Nom	Description
Alerte porte d'entrée	Entrance door is open and is unattended for 5 minutes
Réchauffer un plat surgelé	Freezer gets opened and stove gets turned on within 10 minutes or Freezer gets opened during stove is on , during lunch time
Dépendance de présence	The cupboard gets opened in the kitchen, while a presence in the kitchen is false
Échec de communication	A sensor fails to communicate for 24 hours

Commonalités :

Mesures d'environnement (physique et numérique).

Variabilités :

Contextes dans l'assistance domiciliaire

Nom	Description
Alerte porte d'entrée	Entrance door is open and is unattended for 5 minutes
Réchauffer un plat surgelé	Freezer gets opened and stove gets turned on within 10 minutes or Freezer gets opened during stove is on , during lunch time
Dépendance de présence	The cupboard gets opened in the kitchen, while a presence in the kitchen is false
Échec de communication	A sensor fails to communicate for 24 hours

Commonalités :

Mesures d'environnement (physique et numérique).

Évènements.

Variabilités :

Contextes dans l'assistance domiciliaire

Nom	Description
Alerte porte d'entrée	Entrance door is open and is unattended for 5 minutes
Réchauffer un plat surgelé	Freezer gets opened and stove gets turned on within 10 minutes or Freezer gets opened during stove is on, during lunch time
Dépendance de présence	The cupboard gets opened in the kitchen, while a presence in the kitchen is false
Échec de communication	A sensor fails to communicate for 24 hours

Commonalités :

Mesures d'environnement (physique et numérique).

Évènements.

États.

Variabilités :

Contextes dans l'assistance domiciliaire

Nom	Description
Alerte porte d'entrée	Entrance door is open and is unattended for 5 minutes
Réchauffer un plat surgelé	Freezer gets opened and stove gets turned on within 10 minutes or Freezer gets opened during stove is on, during lunch time
Dépendance de présence	The cupboard gets opened in the kitchen, while a presence in the kitchen is false
Échec de communication	A sensor fails to communicate for 24 hours

Commonalités :

Mesures d'environnement (physique et numérique).

Évènements.

États.

Variabilités :

Contraintes d'interactions (précédence, chevauchement, durée, ...).

Concepts spécifiques au domaine

Évènement

Changement de valeur pour une entité mesurée, à un instant donné.

État

Valeur persistante durant une période. Commence à un instant donné, termine à un instant donné.

Concepts spécifiques au domaine

Évènement

Changement de valeur pour une entité mesurée, à un instant donné.

État

Valeur persistante durant une période. Commence à un instant donné, termine à un instant donné.

Besoins communs

Couvrir les différents objectifs.

Couvrir l'hétérogénéité des données.

Récurrences des interactions dans le flux d'évènements.

Expressivité temporelle adaptée au domaine.

Réactivité des services.

Implantation domiciliaire du contexte

Freezer gets opened and stove gets turned on within 10 minutes during lunch time.

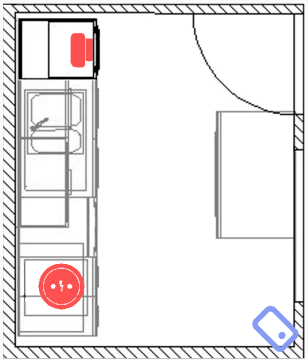
Préparation du déjeuner

Ouvrir la porte du congélateur.


Capteur de contact

Allumer le four.


Capteur de consommation électrique.



Implantation domiciliaire du contexte

Freezer gets opened and stove gets turned on within 10 minutes during lunch time.

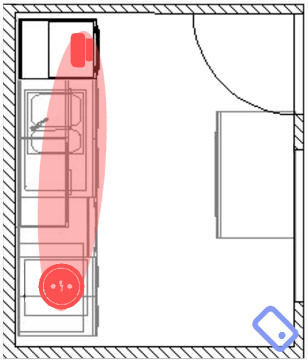
Préparation du déjeuner

Ouvrir la porte du congélateur.

Capteur de contact

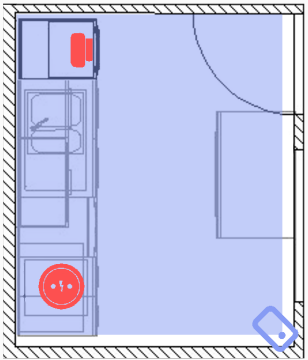
Allumer le four.

Capteur de consommation électrique.



Implantation domiciliaire du contexte

A presence in the Bedroom is true then a presence in the Kitchen is true.



Lever

Présence dans la cuisine.

Capteur de mouvement

Implantation domiciliaire du contexte

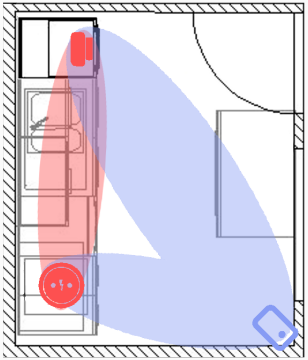
Freezer gets opened and stove gets turned on within 10 minutes during lunch time.

A presence in the Bedroom is true then a presence in the Kitchen is true.

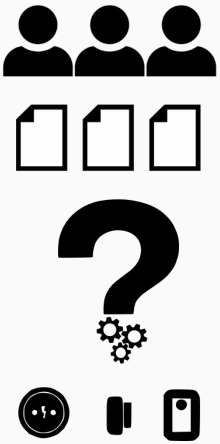
Modèle implicite

Détection de mouvement précède toute autre interaction.

Une présence dans une pièce ne peut pas recouvrir une présence dans une autre pièce.



Synthèse



Concepts spécifiques au domaine.

Expressivité des services.

Comment couvrir les besoins d'expressivité de services sensibles au contexte ?

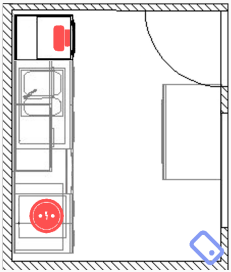
Implantation de la sensibilité au contexte.

Modèle implicite.

Comment rendre explicite le modèle implicite, basé sur les hypothèses ?

Fiabilité de la sensibilité au contexte

Rendre explicite le modèle implicite



Dépendance de présence

Chaque interaction (sauf présence) doit être recouverte par une interaction de présence.

Non ubiquité

Une présence ne peut pas être simultanément détectée dans deux endroits différents.

Rendre explicite le modèle implicite

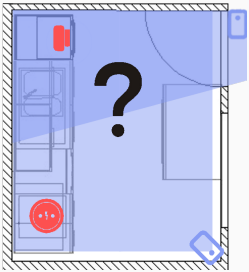
Dépendance de présence

Chaque interaction (sauf présence) doit être recouverte par une interaction de présence.

Non ubiquité

Une présence ne peut pas être simultanément détectée dans deux endroits différents.

Rendre explicite le modèle implicite



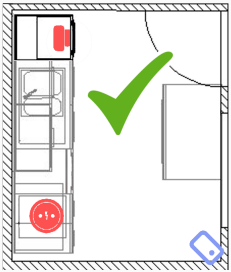
Dépendance de présence

Chaque interaction (sauf présence) doit être recouverte par une interaction de présence.

Non ubiquité

Une présence ne peut pas être simultanément détectée dans deux endroits différents.

Rendre explicite le modèle implicite : Modèle d'infrastructure



Dépendance de présence

Chaque interaction (sauf présence) doit être recouverte par une interaction de présence.

$$\forall \ < k, Kitchen, p > \in \ Log, k \neq Presence \ \Rightarrow$$
$$\exists \ < Presence, Kitchen, p' > \in \ Log, \ p \subseteq p'$$

Non ubiquité

Une présence ne peut pas être simultanément détectée dans deux endroits différents.

$$\forall \ < Presence, l, p > \in \ Log \Rightarrow$$
$$\nexists \ < Presence, l', p' > \in \ Log, \ l \neq l' \wedge \ p' \cap p \neq \varnothing$$

Validation avec le projet pilote de Domassist

Assistance domiciliaire pour personnes âgées

Services définis par des ergothérapeutes, psychologues, experts en vieillissement.

Applications de surveillance des activités du quotidien.

Applications de sécurité du domicile et de l'utilisateur.

Étude conduite avec 24 participants vivant seuls pendant 9 mois.

Name	Rule
Dépendance de présence	$\forall\ < i, l, p > \in\ Log, i \neq Presence\ \Rightarrow$ $\exists\ < Presence, l, p' > \in\ Log,\ p \subseteq p'$
Porte restée ouverte	$\forall\ < Opening, l, p > \in\ Log \Rightarrow \#p < MAX$
Non ubiquité	$\forall\ < Presence, l, p > \in\ Log \Rightarrow$ $\nexists\ < Presence, l', p' > \in\ Log,\ l \neq l' \wedge\ p' \cap p \neq \emptyset$
Taux de déclenchement	$\forall\ < k, l, p > \in\ Log \Rightarrow$ $\#(p - Prev(< k, l, p >)) < MAX$

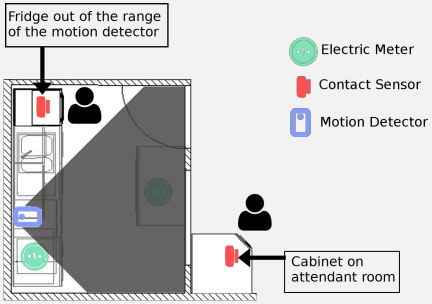
Fiabilité : Résultats

Non-conformités permanentes :

Non-conformités émergentes :

Fiabilité : Résultats

Non-conformités permanentes :



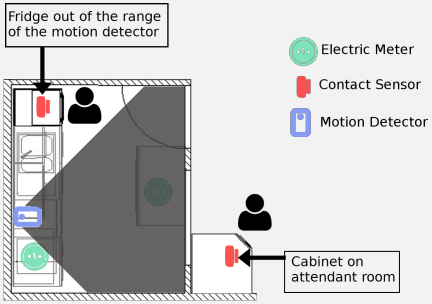
→ Adapter le modèle.

→ Repositionner le capteur.

Non-conformités émergentes :

Fiabilité : Résultats

Non-conformités permanentes :



→ Adapter le modèle.

→ Repositionner le capteur.

Non-conformités émergentes :

Chute du capteur de contact.

Capteur de mouvement défaillant.

→ Intervention d'un technicien.

Synthèse : Assurer la fiabilité du contexte
Modèle d'infrastructure explicite sous forme de règles [CCV16]

Conformité d'une installation avec un modèle

Certifie l'installation.

Guide l'installation.

Application certifiée pour un modèle/installation.

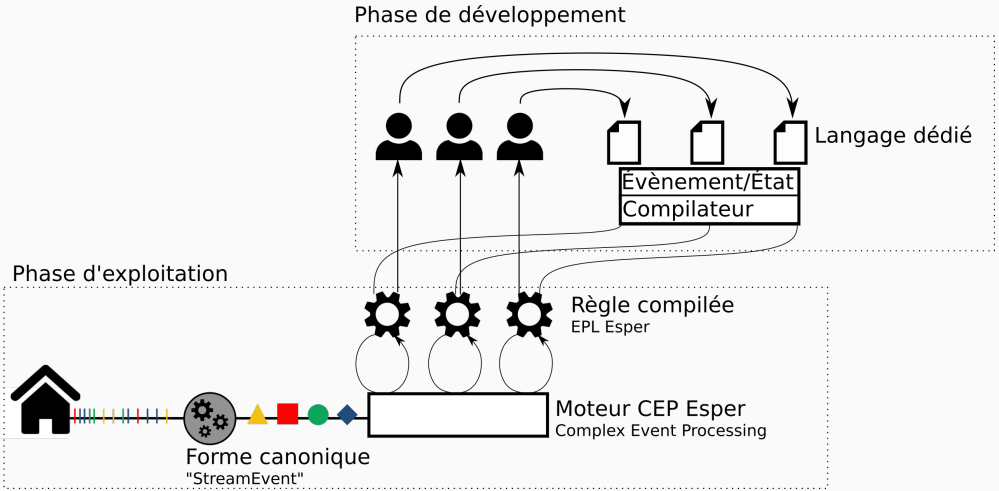
Vérification continue

Détection des pannes.

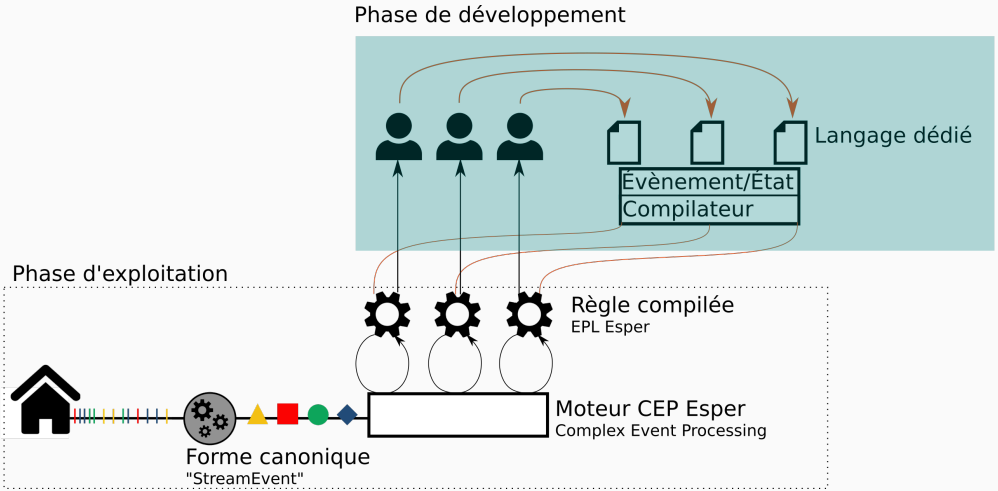
Aide au diagnostic.

Maloya : langage dédié aux services sensibles au contexte

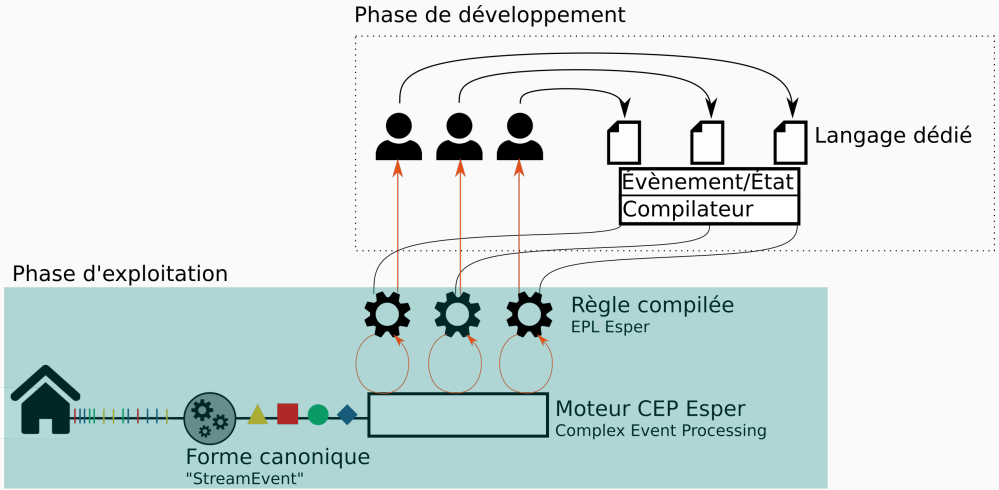
Une approche dédiée à l'assistance domiciliaire



Une approche dédiée à l'assistance domiciliaire



Une approche dédiée à l'assistance domiciliaire



Langage de règles

Freezer gets opened and stove gets turned on within 10 minutes or
Freezer gets opened during stove is on , during lunch time

Langage de règles

Freezer gets opened and stove gets turned on within 10 minutes or
Freezer gets opened during stove is on , during lunch time

Évènement :

```
//Syntaxe textuelle:
freezer becomes open
```

Langage de règles

Freezer gets opened and stove gets turned on within 10 minutes or
Freezer gets opened during stove is on, during lunch time

Évènement :

```
//Syntaxe textuelle:
freezer becomes open
```

État :

```
//Syntaxe textuelle:
stove is on
```

Langage de règles

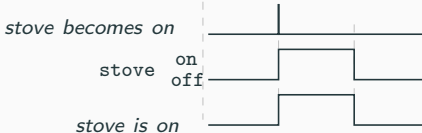
Freezer gets opened and stove gets turned on within 10 minutes or
Freezer gets opened during stove is on, during lunch time

Évènement :

```
//Syntaxe textuelle:
freezer becomes open
```

État :

```
//Syntaxe textuelle:
stove is on
```



Langage de règles

Freezer gets opened **and** stove gets turned on **within 10 minutes** or
Freezer gets opened during stove is on , during lunch time

Opérateur Precedes

```
//Syntaxe textuelle:
( freezer becomes open precedes
within 10 minutes stove becomes on )
```

Langage de règles

Freezer gets opened **and** stove gets turned on **within 10 minutes** or
Freezer gets opened during stove is on , during lunch time

Opérateur Precedes

```
//Syntaxe textuelle:
( freezer becomes open precedes
within 10 minutes stove becomes on )
```

Variantes :

- e_1 **precedes within** t e_2
- e_1 **precedes by** t e_2

Opérateurs

Every time e_1 *immediately* precedes e_2

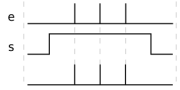
e_1 **precedes** $e_2 \Leftrightarrow Precedes(e_1, e_2)$

Variants :

e_1 **precedes within** t $e_2 \Leftrightarrow Precedes\_less(t)(e_1, e_2)$

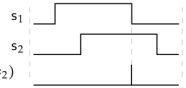
e_1 **precedes by** t $e_2 \Leftrightarrow Precedes\_greater(t)(e_1, e_2)$

Every time e occurs during state s



e **during** s $\Leftrightarrow During(e, s)$

Every time state s_1 overlaps with state s_2



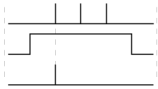
s_1 **overlapping** $s_2 \Leftrightarrow Overlapping(s_1, s_2)$

Variants :

s_1 **overlapping within** t $s_2 \Leftrightarrow Overlapping\_less(t)(s_1, s_2)$

s_1 **overlapping for** t $s_2 \Leftrightarrow Overlapping\_greater(t)(s_1, s_2)$

The first occurrence of event e during state s



e **occurs while** s $\Leftrightarrow Occurs(e, s)$

The first occurrence of state s_1
(partially) superposed with state s_2

s_1 **occurs while** $s_2 \Leftrightarrow Occurs(s_1, s_2)$

Variants :

s_1 **occurs within** t **while** $s_2 \Leftrightarrow Occurs\_less(t)(s_1, s_1)$

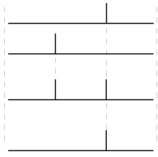
s_1 **occurs for** t **while** $s_2 \Leftrightarrow Occurs\_greater(t)(s_1, s_1)$

Trigger whenever any of the events happens

$\{e_1$ **or** $\ldots$ **or** $e_n\} \Leftrightarrow Or(e_1, \ldots, e_n)$

Trigger as soon as every event happens

$\{e_1$ **and** $\ldots$ **and** $e_n\} \Leftrightarrow And(e_1, \ldots, e_n)$



Opérateurs

Every time e_1 *immediately* precedes e_2

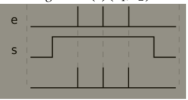
e_1 **precedes** $e_2 \Leftrightarrow Precedes(e_1, e_2)$

Variants :

e_1 **precedes within** t $e_2 \Leftrightarrow Precedes\_less(t)(e_1, e_2)$

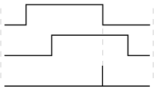
e_1 **precedes by** t $e_2 \Leftrightarrow Precedes\_greater(t)(e_1, e_2)$

Every time e occurs during state s



e **during** s $\Leftrightarrow During(e, s)$

Every time state s_1 overlaps with state s_2



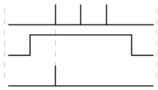
s_1 **overlapping** $s_2 \Leftrightarrow Overlapping(s_1, s_2)$

Variants :

s_1 **overlapping within** t $s_2 \Leftrightarrow Overlapping\_less(t)(s_1, s_2)$

s_1 **overlapping for** t $s_2 \Leftrightarrow Overlapping\_greater(t)(s_1, s_2)$

The first occurrence of event e during state s



e **occurs while** s $\Leftrightarrow Occurs(e, s)$

The first occurrence of state s_1
(partially) superposed with state s_2

s_1 **occurs while** $s_2 \Leftrightarrow Occurs(s_1, s_2)$

Variants :

s_1 **occurs within** t **while** $s_2 \Leftrightarrow Occurs\_less(t)(s_1, s_1)$

s_1 **occurs for** t **while** $s_2 \Leftrightarrow Occurs\_greater(t)(s_1, s_1)$

Trigger whenever any of the events happens

$\{e_1$ **or** $\ldots$ **or** $e_n\} \Leftrightarrow Or(e_1, \ldots, e_n)$

Trigger as soon as every event happens

$\{e_1$ **and** $\ldots$ **and** $e_n\} \Leftrightarrow And(e_1, \ldots, e_n)$

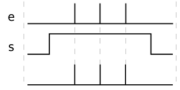
Opérateurs

Every time e_1 *immediately* precedes e_2

e_1 **precedes** $e_2 \Leftrightarrow Precedes(e_1, e_2)$

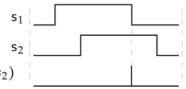
Variants : e_1 **precedes within** t $e_2 \Leftrightarrow Precedes\_less(t)(e_1, e_2)$
e_1 **precedes by** t $e_2 \Leftrightarrow Precedes\_greater(t)(e_1, e_2)$

Every time e occurs during state s



e **during** s $\Leftrightarrow During(e, s)$

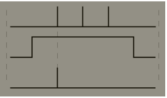
Every time state s_1 overlaps with state s_2



s_1 **overlapping** $s_2 \Leftrightarrow Overlapping(s_1, s_2)$

Variants : s_1 **overlapping within** t $s_2 \Leftrightarrow Overlapping\_less(t)(s_1, s_2)$
s_1 **overlapping for** t $s_2 \Leftrightarrow Overlapping\_greater(t)(s_1, s_2)$

The first occurrence of event e during state s



e **occurs while** s $\Leftrightarrow Occurs(e, s)$

The first occurrence of state s_1
(partially) superposed with state s_2

s_1 **occurs while** $s_2 \Leftrightarrow Occurs(s_1, s_2)$

Variants : s_1 **occurs within** t **while** $s_2 \Leftrightarrow Occurs\_less(t)(s_1, s_1)$
s_1 **occurs for** t **while** $s_2 \Leftrightarrow Occurs\_greater(t)(s_1, s_1)$

Trigger whenever any of the events happens
$\{e_1$ **or** $\ldots$ **or** $e_n\} \Leftrightarrow Or(e_1, \ldots, e_n)$

Trigger as soon as every event happens
$\{e_1$ **and** $\ldots$ **and** $e_n\} \Leftrightarrow And(e_1, \ldots, e_n)$

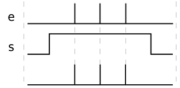
Opérateurs

Every time e_1 *immediately* precedes e_2

e_1 **precedes** $e_2 \Leftrightarrow Precedes(e_1, e_2)$

Variants :
e_1 **precedes within** t $e_2 \Leftrightarrow Precedes\_less(t)(e_1, e_2)$
e_1 **precedes by** t $e_2 \Leftrightarrow Precedes\_greater(t)(e_1, e_2)$

Every time e occurs during state s



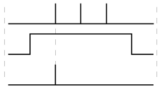
e **during** s $\Leftrightarrow During(e, s)$

Every time state s_1 overlaps with state s_2

s_1 **overlapping** $s_2 \Leftrightarrow Overlapping(s_1, s_2)$

Variants :
s_1 **overlapping within** t $s_2 \Leftrightarrow Overlapping\_less(t)(s_1, s_2)$
s_1 **overlapping for** t $s_2 \Leftrightarrow Overlapping\_greater(t)(s_1, s_2)$

The first occurrence of event e during state s



e **occurs while** s $\Leftrightarrow Occurs(e, s)$

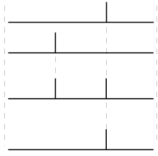
The first occurrence of state s_1
(partially) superposed with state s_2

s_1 **occurs while** $s_2 \Leftrightarrow Occurs(s_1, s_2)$

Variants :
s_1 **occurs within** t **while** $s_2 \Leftrightarrow Occurs\_less(t)(s_1, s_1)$
s_1 **occurs for** t **while** $s_2 \Leftrightarrow Occurs\_greater(t)(s_1, s_1)$

Trigger whenever any of the events happens
{e_1 **or** ... **or** e_n} $\Leftrightarrow Or(e_1, \ldots, e_n)$

Trigger as soon as every event happens
{e_1 **and** ... **and** e_n} $\Leftrightarrow And(e_1, \ldots, e_n)$



Définition d'un service

Syntaxe textuelle

```
{( freezer becomes open  precedes within 10 minutes  stove becomes on )
   or
  ( freezer becomes open  occurs while  stove is on )
} occurs while  lunchTime
```

Syntaxe abstraite

```
Occurs ( Or (
  Precedes_less(10min) ( freezer => open , stove => on ),
  Occurs ( freezer => open , stove = on )),
lunchTime )
```

Définition d'un service

Syntaxe textuelle

```
{( freezer becomes open  precedes within 10 minutes  stove becomes on )
   or
  ( freezer becomes open  occurs while  stove is on )
} occurs while  lunchTime
```

Syntaxe abstraite

```
Occurs ( Or (
  Precedes_less(10min) ( freezer => open , stove => on ),
  Occurs ( freezer => open , stove = on )),
lunchTime )
```

Définition d'un service

Syntaxe textuelle

```
{( freezer becomes open  precedes within 10 minutes  stove becomes on )
   or
  ( freezer becomes open  occurs while  stove is on )
} occurs while  lunchTime
```

Syntaxe abstraite

```
Occurs (  Or (
  Precedes_less(10min) ( freezer => open , stove => on ),
  Occurs ( freezer => open , stove = on )),
lunchTime )
```

Définition d'un service

Syntaxe textuelle

```
{( freezer becomes open precedes within 10 minutes stove becomes on )
  or
  ( freezer becomes open occurs while stove is on )
} occurs while lunchTime
```

Syntaxe abstraite

```
Occurs ( Or (
  Precedes_less(10min) ( freezer => open , stove => on ),
  Occurs ( freezer => open , stove = on )),
lunchTime )
```

Étapes de compilation

Syntaxe abstraite :

```
Occurs( Precedes _ less(10min) (freezer=>open,
                                stove=>on ),
        lunchTime )
```

Étapes de compilation : Vers Pseudo-code EPL

Syntaxe abstraite :

```
Occurs( Precedes _ less(10min) (freezer=>open,
                                stove=>on ),
        lunchTime )
```

Pseudo-code EPL :

```
//Window1
( every freezer => open ->  stove => on
  and not (freezer=>open) where timer:within(10min) )

every  lunchTime=>begin  ->
   Window1 (timestamp > (lunchTime=>begin).timestamp)
   and not lunchTime=>end
```

Étapes de compilation : Vers Pseudo-code EPL

Gestion des états

Syntaxe abstraite :

```
Occurs( Precedes _ less(10min) (freezer=>open,
                                stove=>on ),
        lunchTime )
```

Pseudo-code EPL :

```
//Window1
( every freezer => open ->  stove => on
  and not (freezer=>open) where timer:within(10min) )

every lunchTime=>begin ->
   Window1 (timestamp > (lunchTime=>begin).timestamp)
   and not lunchTime=>end
```

Étapes de compilation : Vers Pseudo-code EPL

Gestion des états , Composition

Syntaxe abstraite :

```
Occurs( Precedes _ less(10min) (freezer=>open,
                                stove=>on ),
        lunchTime )
```

Pseudo-code EPL :

```
//Window1
( every freezer => open ->  stove => on
  and not (freezer=>open) where timer:within(10min) )

every lunchTime=>begin ->
  Window1 (timestamp > (lunchTime=>begin).timestamp)
  and not lunchTime=>end
```

Étapes de compilation : Vers Pseudo-code EPL

Gestion des états , Composition , Timer explicites

Syntaxe abstraite :

```
Occurs( Precedes _ less(10min) (freezer=>open,
                                stove=>on ),
        lunchTime )
```

Pseudo-code EPL :

```
//Window1
( every freezer => open ->  stove => on
  and not (freezer=>open) where timer:within(10min) )

every lunchTime=>begin ->
  Window1 (timestamp > (lunchTime=>begin).timestamp)
  and not lunchTime=>end
```

Étapes de compilation : Vers EPL Esper

Gestion des états , Composition , Timer explicites

Syntaxe abstraite :

```
Occurs( Precedes _ less(10min) (freezer=>open,
                                stove=>on ),
        lunchTime )
```

Pseudo-code EPL :

```
//Window1
( every freezer => open ->  stove => on
  and not (freezer=>open) where timer:within(10min) )

every lunchTime=>begin ->
  Window1 (timestamp > (lunchTime=>begin).timestamp)
  and not lunchTime=>end
```

EPL Esper :

```
create window Wind.std:unique(location,kind,user)
select * from StreamEvent

insert into Wind select arg from pattern [
  (every arg=StreamEvent(location='Kitchen',
                         kind='Freezer',status='open') ->
     StreamEvent(location='Kitchen',kind='Stove',
                 status='on',user=arg.user)
     and not (StreamEvent(location='Kitchen',kind='Freezer',
                          status='open',user=arg.user))
     where timer:within (10min) ) ]

select Cal_L_b,arg from pattern [
  every Cal_L_b= StreamEvent(location='Lunch',kind='Calendar',
                             status!='end') ->
    arg=Wind(timestamp>Cal_L_b.timestamp,user=Cal_L_b.user)
    and not StreamEvent(location='Lunch',kind='Calendar',
                        status='end',user=Cal_L_b.user) ]
```

Étapes de compilation : Vers EPL Esper

Gestion des états , Composition , Timer explicites , évènements

Syntaxe abstraite :

```
Occurs( Precedes _ less(10min) (freezer=>open,
                                stove=>on ),
        lunchTime )
```

Pseudo-code EPL :

```
//Window1
( every freezer => open -> stove => on
  and not (freezer=>open) where timer:within(10min) )

every lunchTime=>begin ->
  Window1 (timestamp > (lunchTime=>begin).timestamp)
  and not lunchTime=>end
```

EPL Esper :

```
create window Wind.std:unique(location,kind,user)
select * from StreamEvent

insert into Wind select arg from pattern [
  (every arg=StreamEvent(location='Kitchen',
                         kind='Freezer',status='open') ->
     StreamEvent(location='Kitchen',kind='Stove',
                 status='on',user=arg.user)
     and not (StreamEvent(location='Kitchen',kind='Freezer',
                          status='open',user=arg.user))
     where timer:within (10min) ) ]

select Cal_L_b,arg from pattern [
  every Cal_L_b= StreamEvent(location='Lunch',kind='Calendar',
                             status!='end') ->
    arg=Wind(timestamp>Cal_L_b.timestamp,user=Cal_L_b.user)
    and not StreamEvent(location='Lunch',kind='Calendar',
                        status='end',user=Cal_L_b.user) ]
```

Validation avec le projet Domassist

Expressivité

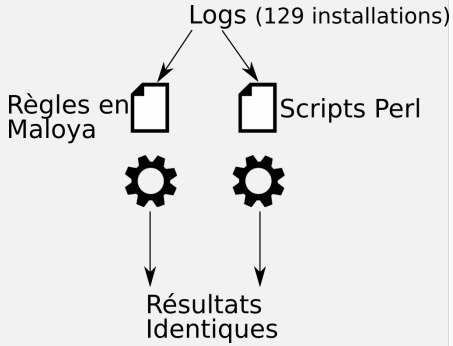
Redéfinition de 55 services.

Validation avec le projet Domassist

Expressivité

Redéfinition de 55 services.

Exactitude



Validation avec le projet Domassist

Expressivité

Redéfinition de 55 services.

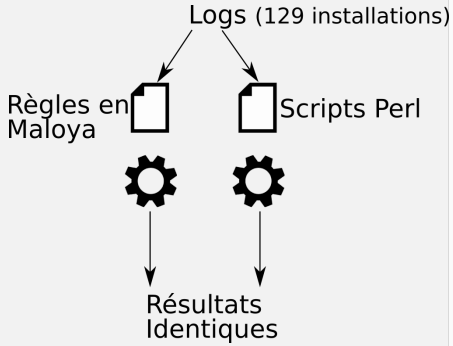
Performance

Latence inférieure à une seconde.

Occupation mémoire de 352Mo en moyenne.

55 règles, 129 installations, H24, 1 mois

Exactitude



Conclusion

Conclusion : couvrir les besoins des services sensibles au contexte

Langage dédié :

- Définition de services unifiée.
- Expressivité couvre les objectifs de services.
- Modèle d'infrastructure exprimé avec des services.

Compilation masquant la complexité du traitement événementiel :

- Services plus facile à exprimer et comprendre.

Perspectives

Définition de services par les intervenants :

- Étude ergonomique en cours (compréhension).
- Langage graphique (programmation).

Cibles de compilation :

- Stream processing (Apache Spark, Flink).

Évaluer le gain de l'éffort de développement.

Bibliographie

A. Carteron et al. “Improving the Reliability of Pervasive Computing Applications by Continuous Checking of Sensor Readings”. In : *2016 Intl IEEE Conferences on Ubiquitous Intelligence Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld)*. Juillet 2016, p. 41–50. DOI : 10.1109/UIC-ATC-ScalCom-CBDCom-IoP-SmartWorld.2016.0029