ESO Classes: Definitions, Class mappings, Role Mappings, Assertions and Examples of the Instantiation of the Assertions.

This file provides a human readable version of the Event and Situation Ontology 1.0., developed for the NewsReader project (www.newsreader-project.eu).

All classes are in alphabetical order. For each class we provide:
-the subclass relation
-the class definition
-the mappings from ESO classes to FrameNet and SUMO (as available online at June 20, 2015)
-the mappings from ESO roles to FrameNet Frame Elements
-the assertions for each class defining the situation that holds before, after and/or during the event (in a non-formal transcription).
-examples that show what the ESO class assertions can infer from a sentence annotated with FrameNet-based SRL.

For the class eso:Damaging, we also provide a commented full OWL and RDF version that shows the existential restriction for relative values and examples of the assertion instantiations.
This example can be found at the end of this document.

Date: June 24th 2015

For questions and remarks, please contact:
r.h.segers@vu.nl

ESO CLASSES IN ALPHABETICAL ORDER:

-**Arriving** subclassOf:Translocation

"The subclass of Translocation where someone or something arrives at a location."

Class mappings:
closeMatch: fn:Arriving
closeMatch: fn:Vehicle_landing
closeMatch: sumo:Arriving

For the roles and assertions and, see: Translocation.

EXAMPLES:

"Mary approached the White House with a grim face."

pre situation	Mary	notAtPlace	the White House
post situation	Mary	atPlace	the White House

"Mary arrived in Washington from Dulles National Airport."

pre situation	Mary	atPlace	Dulles National Airport
	Mary	notAtPlace	Washington
post situation	Mary	atPlace	Washington
	Mary	notAtPlace	Dulles National Airport

-**Attacking** subclassOf: IntentionalEvent

"The subclass of IntentionalEvent where someone or something is assaulted with the intention to cause some harm."

Class mappings:
closeMatch: fn:Attack
closeMatch: sumo:ViolentContest

Role mappings:
damaging-undergoer: fn: Object, fn:Victim, fn: Experiencer, fn:Body_part,
fn: Patient, fn: Artifact
damaging-state-1: - (blank node)
damaging-state-2: - (blank node)
damaging-damage: -
activity: -

Assertions:

pre situation:	damaging-undergoer	inState	damaging-state-1
	damaging-state-1	hasRelativeValue	"+"
post situation:	damaging-undergoer	inState	damaging-state-2
	damaging-state-2	hasRelativeValue	"-"
	damaging-undergoer	isDamaged	true
	damaging-undergoer	hasDamage	damaging-damage
	damaging-damage	hasNegativeEffectOn	activity

Note that the last two assertions will not be instantiated as no FrameNet roles exist for the ESO roles damaging-damage and activity.
Note that damaging-state-1 and damaging-state-2 are modeled with an existential restriction that allows to create a blank node in the named graph.

EXAMPLES:

"Marie attacked John with a knife."

pre situation	John	inState	:xyz123
	:xyz123	hasRelativeValue	+
post situation	John	inState	:xyz124
	:xyz124	hasRelativeValue	-
	John	isDamaged	true

"The army bombed the power plant."

pre situation	the power plant	inState	:xyz125
	xyz125	hasRelativeValue	+
post situation	the power plant	inState	:xyz126
	:xyz126	hasRelativeValue	-
	the power plant	isDamaged	true

"The hurricane struck West-Virginia."

pre situation	West-Virginina	inState	:abc123
	:abc123	hasRelativeValue	+
post situation	West-Virginia	inState	:abc124
	:abc124	hasRelativeValue	-
	West-Virginia	isDamaged	true

-BeginningARelationship subclassOf: IntentionalEvent
"The subclass of IntentionalEvent were people start or form a personal relationship with each other".

Class mappings:
broadMatch: fn:Forming_relationships

Role mappings:
relationship-partner-1: fn:Partner_1
relationship-partner-2: fn:Partner_2

relationship-partners: fn:Partner_1, fn:Partner_2, fn:Partners

<u>Assertions:</u>

pre situation	relationship-partner-1	notInRelationshipWith	relationship-partner-2
	relationship-partners	inRelationship	false
post situation	relationship-partner-1	inRelationshipWith	relationship-partner-2
	relationship-partners	inRelationship	true

EXAMPLES:

"John married Mary in 2011."

pre situation	John	notInRelationshipWith	Mary
	John, Mary	inRelationship	false
post situation	John	inRelationshipWith	Mary
	John, Mary	inRelationship	true

"The secret wedding of John and Mary!"

pre situation	John and Mary	inRelationship	false
post situation	John and Mary	inRelationship	true

"John married again in 2014."

pre situation	John	inRelationship	false
post situation	John	inRelationship	true

-BeingAtAPlace subclassOf: StaticEvent
"Static event where some entity is at a location."

<u>Class mappings:</u>
closeMatch: fn:Residence
closeMatch: fn:Presence
closeMatch: fn:Temporary_stay
closeMatch: fn:Being_located

<u>Role mappings:</u>
atPlace-theme: fn:Theme, fn:Resident, fn:Entity, fn:Guest.
atPlace-location: fn:Location

<u>Assertions:</u>

during situation:	atPlace-theme	atPlace	atPlace-location

EXAMPLES:

"Marie stayed at the Hilton Hotel."

during situation	Marie	atPlace	Hilton Hotel

"Oil reservoirs are present in Rotterdam."

during situation	oil reservoirs	atPlace	Rotterdam

"John lives in Amsterdam."

during situation	John	atPlace	Amsterdam

"John is the first resident at King's Landing."

during situation	John	atPlace	King's Landing

-BeingDamaged subclassOf: StaticEvent

"Static event where some entity is in a damaged state."

<u>Class mappings:</u>
broadMatch: fn:Being_operational

<u>Role mappings:</u>
damaging_undergoer: fn:Object, fn:Victim, fn: Experiencer, fn:Body_part, fn: Patient, fn: Artifact.
damaging-damage: -
activity: -

<u>Assertions:</u>

during-situation:	damaging-undergoer	isDamaged	true
	damaging-undergoer	hasDamage	damaging-damage
	damaging-damage	hasNegativeEffectOn	activity

Note that the last two assertions will not be instantiated as no FrameNet roles exist for the ESO roles damaging-damage and activity.

EXAMPLE:

"The suspension of this car is broken."

during-situation

the suspension of this car	isDamaged	true
(this car	hasDamage	broken suspension)
(broken suspension	hasNegativeEffectOn	operating)

-BeingEmployed subclassOf: StaticEvent

"Static event where someone is working in a position and is compensated for her work by some form of payment."

<u>Class mappings:</u>
closeMatch: fn:Being_employed
closeMatch: fn:Employing

<u>Role mappings:</u>
employment-employee: fn:Employee
employment-employer: fn:Employer
employment-function: fn:Position
employment-value: fn:Compensation
employment-task: fn:Task
employment-attribute: -

<u>Assertions:</u>

during situation	employment-employee	employedAt	employment-employer
	employment-employee	hasFunction	employment-function
	employment-employee	hasTask	employment-task
	employment-employee	hasAttribute	employment-attribute
	employment-attribute	hasValue	employment-value
	employment-employee	isEmployed	true

Note that employment-attribute is modeled with an existential restriction that allows to create a blank node in the named graph.

EXAMPLES:

"Ford employed Marie as CFO."

during situation	Marie	employedAt	Ford
	Marie	hasFunction	CFO
	Marie	isEmployed	true

"Marie works as CFO for 2000 dollar a month."

during situation	Marie	hasFunction	CFO
	Marie	hasAttribute	:xyz667
	:xyz667	hasValue	2000 dollar
	Marie	isEmployed	true

"Marie is employed at Ford to handle the severe financial issues."

during situation	Marie	employedAt	Ford
	Marie	hasTask	to handle the severe financial issues
	Marie	isEmployed	true

-BeingInAPersonalRelationship subclassOf:StaticEvent

"The subclass of StaticEvent where persons are in some personal relationship."

Class mappings:
closeMatch: fn:Personal_relationship

Role mappings:
relationship-partner-1: fn:partner_1
relationship-partner-2: fn:partner_2
relationship-partners: fn:partners, fn: partner_1, fn: partner_2

Assertions:

during situation	relationship-partner-1	inRelationshipWith	relationship-partner-2
during situation	relationship-partners	inRelationship	true

EXAMPLES:

"John dates Marie."

during-situation	John	inRelationshipWith	Marie
	John, Marie	inRelationship	true

"John is married to Marie."

during situation	John	inRelationshipWith	Marie
	John, Marie	inRelationship	true

-BeingInExistence subclassOf: StaticEvent

“Static event where some entity exists.”

Class mappings:
closeMatch: fn:Existence

Role mappings:
exist-theme: fn:Entity

Assertions:

during situation	exist-theme	exist	true

EXAMPLES:

"Cars with a Wankel engine still exist."

during situation	cars with a Wankel engine	exist	true

"There were human settlements near the volcano."

during situation	human settlements near the volcano	exist	true

-BeingInUse subclassOf StaticEvent

"The static event class where something is in use by an agent
(in some particular role or for some purpose)."

Class mappings:
closeMatch: fn:Using
closeMatch: fn:UsingResource
broadMatch: fn:BeingOperational

Role mappings:
inuse-entity-1: fn:Agent
inuse-entity-2 fn:Instrument, fn:Resource, fn:Object
inuse-function: fn:Role
inuse-purpose: fn:Purpose

Assertions:

during situation	inuse-entity-1	uses	inuse-entity-2
	inuse-entity-2	hasFunction	inuse-function
	inuse-entity-2	hasPurpose	inuse-purpose
	inuse-entity-2	inFunction	true

"Ford uses codename X for operations in India."

during situation	Ford	uses	codename X
	codename X	hasPurpose	operations in India
	codename X	inFunction	true

"Ford used codename X name as cover."

during situation	Ford	uses	operational name
	codename X	hasFunction	cover
	codename X	inFunction	true

"Mary used her Peugeot 205 to drive to work."

during situation	Mary	uses	her Peugeot 205
	her Peugeot 205	hasPurpose	drive to work
	her Peugeot 205	inFunction	true

"The system works."

during situation	the system	inFunction	true

-BeingLeader subclassOf: StaticEvent

“StaticEvent where someone is leader of some group of persons or organization.”

Class mappings:
closeMatch: fn:Leadership

Role mappings:
leader-entity: fn:Leader
leader-governed-entity: fn:Governed
leader-function: fn:Role

Assertions:

during situation:	leader-entity	isLeader	true
	leader-entity	isLeaderOf	leader-governed_entity
	leader-entity	hasFunction	leader-function

EXAMPLES:

"John chairs the committee"

during situation	John	isLeader	true
	John	isLeaderOf	the committee

"John ruled over Apple as a king"

during situation	John	isLeader	true
	John	isLeaderOf	Apple
	John	hasFunction	king

"Ford is setting up an operation which is headed by Mary as general manager"

during situation	Mary	isLeader	true
	Mary	hasFunction	general manager

"John is chairman of the committee."

during situation	John	isLeader	true
	John	isLeaderOf	the committee

-BeingOperational subclassOf: StaticEvent
Static event where some device is in function.

Class mappings:
closeMatch: fn:Being-operational

Role mappings:
operational-theme: fn:Object

Assertions:

during situation	operational-theme	inFunction	true

EXAMPLES:

"The new welding power supply works."

during situation	the new welding power supply	inFunction	true

"The new welding power supply is functional."

during situation	the new welding power supply	inFunction	true

-Borrowing subclassOf: Getting

"The subclass of Getting where a person gets something in possession for some period of time after which the item should be given back."

<u>Class mappings:</u>
closeMatch: fn:Borrowing
closeMatch: fn:Borrowing

For the roles and assertions, see: ChangeOfPossession.

EXAMPLE:

"Mary borrowed the car from John"

pre situation	John	hasInPossession	the car
	Marie	notHasInPossession	the car
post situation	John	notHasInPossession	the car
	Marie	hasInPossession	the car

-Buying subclassOf: FinancialTransaction

The subclass of FinancialTransaction where some entity changes of ownership in exchange for money. Note that the buyer is not necessarily the new owner of the entity.

<u>Class mappings:</u>
closeMatch: fn:Commerce_buy
closeMatch: sumo:Buying

For the roles and assertions, see: ChangeOfPossession.

EXAMPLES:

"John bought the flowers for 10 dollar."

pre situation	John	hasInPossession	10 dollar
	John	notHasPossession	the flowers
post situation	John	hasInPossession	the flowers
	John	notHasInPossession	10 dollar
during situation	the flowers	hasValue	10 dollar

"John bought the flowers from Mary."

pre situation	John	notHasInPossession	the flowers
	Mary	hasInPossession	the flowers
post situation	John	hasInPossession	the flowers
	Mary	notHasInPossession	the flowers

"John bought the flowers for Mary."

pre situation	John	notHasInPossession	flowers
	Mary	notHasInPossession	flowers
post situation	John	hasInPossession	flowers
	Mary	hasInPossession	flowers*

**Note that Mary is the 'Recipient' in FrameNet. While this FrameNet role is important for some subclasses of eso: ChangeOfPossession, for eso:Buying, this role is less prominent. However, the roles and assertions for this sub hierarchy are modeled at the highest possible level in the ontology (ChangeOfPossession) and are inherited by e.g. Buying.*
As a result, in some cases the assertions of the post situation of Buying can generate a questionable statement.

-ChangeOfPossession subclassOf: DynamicEvent

"The subclass of DynamicEvent where some entity changes possession. Note that this often but not necessarily implies a change of location of the entity."

Class mappings:
relatedMatch: fn:Transfer
closeMatch: sumo: ChangeOfPossession

Role mappings:
possession-owner_1: fn:Supplier, fn:Exporter, fn:Donor, fn:Victim, fn:Source, fn:Lender, fn:Exporting_area, fn:Sender, fn:Seller
possession-owner_2: fn:Perpetrator, fn:Importing_area, fn:Importer, fn:Lessee, fn:Buyer, fn:Recipient, fn:Borrower, fn:Agent
possession-theme: fn:Theme, fn:Goods, fn:Possession

Assertions:

pre situation	possession-owner_1	hasInPossession	possession-theme
	possession-owner_2	notHasInPossession	possession-theme
post situation	possession-owner_1	notHasInPossession	possession-theme
	possession-owner_2	hasInPossession	possession-theme

EXAMPLES:

"Marie stole the car keys from John"

pre situation	John	hasInPossession	car keys
	Marie	notHasInPossession	car keys
post situation	John	notHasInPossession	car keys
	Marie	hasInPossession	car keys

"Ford exported 3000 cars to India last month"

pre situation	Ford	hasInPossession	3000 cars
	India	notHasInPossession	3000 cars
post situation	Ford	notHasInPossession	3000 cars
	India	hasInPossession	3000 cars

-ChangingShape subclassOf:InternalChange

"The subclass of InternalChange where the shape of an entity is changed."

Class mappings:
closeMatch: fn:Manipulate_into_shape
closeMatch: fn:Reshaping
closeMatch: sumo:ShapeChange

Role mappings:
changingshape-entity: fn:Undergoer, fn:Theme
changingshape-initialshape: -
changingshape-finalshape: fn:Configuration, fn:Resultant_configuration, fn:Result

Assertions:

pre situation	changingshape-entity	inState	changingshape-initialshape
	changingshape-entity	notInState	changingshape-finalshape
post situation	changingshape-entity	inState	changingshape-finalshape
	changingshape-entity	notInState	changingshape-initialshape

Note that changingshape-initialshape and changingshape-finalshape are modeled with an existential restriction that allows to create a blank node in the named graph.

EXAMPLES:

"John moulded the paste into a ball."

pre situation	the paste	inState	:xyz130
	the paste	notInState	ball
post situation	the paste	inState	ball
	the paste	notInState	:xyz130

"John folded the paper."

pre situation	the paper	inState	:xyz134
	the paper	notInState	:abc123
post situation	the paper	inState	:abx123
	the paper	notInState	:xyz134

-Collaboration subclassOf: StaticEvent

"Static event where people work together for some period of time."

Class mappings:
closeMatch: fn:Collaboration
closeMatch: sumo:Cooperation

Role mappings:
collaboration-partner-1: fn:Partner_1
collaboration-partner-2: fn:Partner_2
collaboration-partners: fn:Partner_1, fn:Partner_2, fn:Partners
collaboration-project: fn:Undertaking

Assertions:

during situation	collaboration-partner-1	collaboratesWith	collaboration-partner-2
	collaboration-partners	inCollaboration	true
	collaboration-partners	hasProject	collaboration-project

EXAMPLES:

"John collaborates with Mary on a book."

during situation	John	collaboratesWith	Mary
	John, Mary	hasProject	a book
	John, Mary	inCollaboration	true

"The left wing parties are conspiring to impeach the president."

during situation	the left wing parties	hasProject	to impeach the president
	the left wing parties	inCollaboration	true

-Creating subclassOf: InternalChange

"The subclass of InternalChange where something is made, created, build, constructed, etc."

Class mappings:
closeMatch: fn:Building
closeMatch: fn:Intentionally_create
closeMatch: fn:Creating
closeMatch: fn:Manufacturing
closeMatch: sumo:Constructing
closeMatch: sumo:Creation
closeMatch: sumo:Manufacture
closeMatch: sumo:Making

Role mappings:

creating-theme: fn: Product, fn:Created_entity

Assertions:

pre situation	creating-theme	exist	false
post situation	creating-theme	exist	true

EXAMPLES:

"The company was founded in 1981."

pre situation	the company	exist	false
post situation	the company	exist	true

"Rover assembled 22.000 Morris Minis from 1986 onwards."

pre situation	22.000 Morris Minis	exist	false
post situation	22.000 Morris Minis	exist	true

"Mary builds a new house on the hill."

pre situation	a new house on the hill	exist	false
post situation	a new house on the hill	exist	true

-Damaging subclassOf: InternalChange

"The subclass of InternalChange where something is damaged."

Class mappings:
closeMatch: fn:Render_nonfunctional, fn:Damaging
closeMatch: sumo:Damaging

Role mappings:
damaging-undergoer: fn: Object, fn:Victim, fn: Experiencer, fn:Body_part, fn: Patient, fn: Artifact
damaging-state-1: -
damaging-state-2: -
damaging-damage: -
activity: -

Assertions:

pre situation:	damaging-undergoer	inState	damaging-state-1
	damaging-state-1	hasRelativeValue	"+"
post situation:	damaging-undergoer	inState	damaging-state-2
	damaging-state-2	hasRelativeValue	"-"
	damaging-undergoer	isDamaged	true
	damaging-undergoer	hasDamage	damaging-damage
	damaging-damage	hasNegativeEffectOn	activity

Note that the last two assertions will not be instantiated as no FrameNet roles exist for the ESO roles 'damaging-damage' and 'activity'.
Note that damaging-state1 and damaging-state-2 have an existential restriction that allows to create a blank node in the named graph.

EXAMPLES:

"Marie dented the car"

pre situation	car	inState	:abc123
	:abc123	hasRelativeValue	+
post situation	car	inState	:xyz556
	:xyz556	hasRelativeValue	-
	car	isDamaged	true

"John incapacitated the aircraft."

pre situation	the aircraft	inState	:efg123
	:efg123	hasRelativeValue	+
post situation	the aircraft	inState	:efg345
	:efg345	hasRelativeValue	-
	the aircraft	isDamaged	true

-Decreasing subclassOf: QuantityChange

"The subclass of QuantityChange where some physical quantity or value is decreased."

Class mappings:
broadMatch: fn:Change_of_quantity_of_possession
broadMatch: fn:Cause_change_of_position_on_a_scale
broadMatch: fn:Change_position_on_a_scale
broadMatch: fn:Proliferating_in_number
broadMatch: fn: Expansion
broadMatch: fn: Cause_expansion
closeMatch: sumo:Decreasing

Role mappings:
quantity-item: fn:Item, fn:Possession, fn:Set
quantity-attribute: fn:Attribute, fn:Dimension
quantity-ratio: fn:Size_change, fn:Difference
quantity-value_1: fn:Initial_value, fn:Initial_number, fn:Initial_size, fn:Value_1
quantity-value_2: fn:Final_value, fn:Final_number, fn:Value_2, fn:Result_size

Assertions:

pre situation	quantity-item	hasAttribute	quantity-attribute
	quantity-attribute	hasRelativeValue	+
	quantity-attribute	hasValue	quantity-value_1
post situation	quantity-item	hasAttribute	quantity-attribute
	quantity-attribute	hasRelativeValue	-
	quantity-attribute	hasValue	quantity-value_2
	quantity-item	hasRelativeDecrease	quantity-ratio

Note that quantity-attribute is modeled with an existential restriction that allows to create a blank node in the named graph.

EXAMPLES:

"Ford decreased the production with 2%."

pre situation	production	hasAttribute	:qwe123
	:qwe123	hasRelativeValue	+
post situation	production	hasAttribute	:qwe123
	:qwe123	hasRelativeValue	-
	production	hasRelativeDecrease	2%

"Apple lowered the price of the Iphone from 600 to 500 dollar."

pre situation	Iphone	hasAttribute	price
	price	hasRelativeValue	+
	price	hasValue	600
post situation	Iphone	hasAttribute	price
	price	hasRelativeValue	-
	price	hasValue	500

"The profit shrunk dramatically."

pre situation	profit	hasAttribute	:bnm234
	:bnm234	hasRelativeValue	+
post situation	profit	hasAttribute	:bnm234
	:bnm234	hasRelativeValue	-

-Destroying subclassOf: InternalChange
"The subclass of InternalChange where something gets destroyed."

Class mappings:
closeMatch: fn:Cause_to_fragment
closeMatch: fn:Destroying
closeMatch: sumo:Destruction

Role mappings:
destroying-theme: fn:Whole_patient, fn:Executed, fn:Undergoer, fn:Victim

Assertions:

pre situation:	destroying-theme	exist	true
post situation:	destroying-theme	exist	false

EXAMPLES:

"They demolished the Vauxhall factory."

pre situation	the Vauxhall factory	exist	true
post situation	the Vauxhall factory	exist	false

"Mary tore up the license agreement."

pre situation	the license agreement	exist	true
post situation	the license agreement	exist	false

-Distribution subclassOf: Translocation
"The subclass of Translocation where someone or something translocates a physical object from one location to a bigger area."

Class mappings:
closeMatch: fn:Dispersal

For the assertions and role mappings, see: Translocation.

EXAMPLES

"Bats spread the disease across Sudan."

pre situation	the disease	notAtPlace	Sudan
post situation	the disease	atPlace	Sudan

"The engines were mainly distributed in Korea."

pre situation	the engines	notAtPlace	Korea
post situation	the engines	atPlace	Korea

-DynamicEvent This class is the root of the dynamic event class hierarchy.
(no mappings, no assertions)

-EndingARelationship subclassOf: IntentionalEvent

"The subclass of IntentionalEvent were people end a relationship with each other."

<u>Class mappings:</u>
broadMatch: fn:Forming_relationships

<u>Role mappings:</u>
relationship-partner-1: fn:Partner_1
relationship-partner-2: fn:Partner_2
relationship-partners: fn:Partner_1, fn:Partner_2, fn:Partners

pre situation	relationship-partner-1	inRelationshipWith	relationship-partner-2
	relationship-partners	inRelationship	true
post situation	relationship-partner-1	notInRelationshipWith	relationship-partner-2
	relationship-partners	inRelationship	false

EXAMPLES

"Mary split up with John."

pre situation	John	inRelationshipWith	Mary
	John, Mary	inRelationship	true
post situation	John	notInRelationshipWith	Mary
	John, Mary	inRelationship	false

"John divorced in 2013."

pre situation	John	inRelationship	true
post situation	John	inRelationship	false

"The divorce of John and Mary is on the front page of all tabloids!"

pre situation	John and Mary	inRelationship	false
post situation	John and Mary	inRelationship	true

-Escaping subclassOf: Leaving
"The subclass of Leaving where a person leaves an unwanted location."

<u>Class mappings</u>
closeMatch: fn:Escaping
closeMatch: fn:Fleeing
closeMatch: sumo:Escaping

For the assertions and role mappings, see: Translocation.

EXAMPLES:

"John escaped from Alcatraz."

pre situation	John	atPlace	Alcatraz
post situation	John	notAtPlace	Alcatraz

"John fled to the United States."

pre situation	John	notAtPlace	the United States
post situation	John	atPlace	the United States

-Exporting subclassOf: Selling
"The subclass of Selling where goods are exported to another nation

in exchange for money."

Class mappings:
closeMatch: fn:Exporting
closeMatch: sumo:Exporting

For the assertions and role mappings, see: FinancialTransaction

EXAMPLES:

"Ford exported 10.000 cars to India."

pre situation	Ford	hasInPossession	10.000 cars
	India	notHasInPossession	10.000 cars
post situation	Ford	notHasInPossession	10.000 cars
	India	hasInPossession	10.000 cars

"Car exportation to India."

pre situation	India	notHasInPossession	car
post situation	India	hasInPossession	car

-FinancialTransaction: subclassOf: ChangeOfPossession
"The subclass ofChangeOfPossession where some item changes of ownership in exchange for money."

Class mappings:
closeMatch: fn:CommercialTransaction
closeMatch: sumo:FinancialTransaction

Role mappings:
possession-financial-asset: fn:Money

Inherited role mappings:
possession-owner_1: fn:Supplier, fn:Exporter, fn:Donor, fn:Victim, fn:Source, fn:Lender, fn:Exporting_area, fn:Sender, fn:Seller
possession-owner_2: fn:Perpetrator, fn:Importing_area, fn:Importer, fn:Lessee, fn:Buyer, fn:Recipient, fn:Borrower, fn:Agent
possession-theme: fn:Theme, fn:Goods, fn:Possession
possession-financial-asset: fn:Money

Assertions:

pre situation	possession-owner_1	notHasInPossession	poss.-financial-asset
	possession-owner-2	hasInPossession	poss.-financial-asset
post situation	possession-owner_1	hasInPossession	poss.-financial-asset
	possession-owner_2	notHasInPossession	poss.-financial-asset
during situation	possession-theme	hasValue	possession-value

Inherited assertions from ChangeOfPossession:

pre situation	possession-owner_1	hasInPossession	possession-theme
	possession-owner_2	notHasInPossession	possession-theme
post situation	possession-owner_1	notHasInPossession	possession-theme
	possession-owner_2	hasInPossession	possession-theme

EXAMPLES:

"Marie bought the car from John for 600 dollars"

pre situation	Marie	hasInPossession	600 dollar

	Marie	notHasInPossession	the car
	John	hasInPossession	the car
	John	notHasInPossession	600 dollar
post situation	Marie	hasInPossession	the car
	Marie	notHasInPossession	600 dollar
	John	hasInPossession	600 dollar
	John	notHasInPossession	the car
during situation	the car	hasValue	600 dollar

"Mary paid 600 dollar for the car."

pre situation	Mary	notHasInPossession	the car
	Mary	hasInPossession	600 dollar
post situation			
	Mary	hasInPossession	the car
	Mary	notHasInPossession	600 dollar
during situation	the car	hasValue	600 dollar

-Getting subclassOf: ChangeOfPossession

“The subclass of ChangeOfPossession where a person gets or receives some item.”

Class mappings:
closeMatch: fn:Receiving
closeMatch: fn:Getting
closeMatch: sumo:Getting

For the assertions and role mappings, see: ChangeOfPossession.

EXAMPLES:

"Mary received the strategic report from John."

pre situation	John	hasInPossession	the strategic report
	Mary	notHasInPossession	the strategic report
post situation	John	notHasInPossession	the strategic report
	Mary	hasInPossession	the strategic report

"Mary gained the respect of her staff."

pre situation	Mary	notHasInPossession	the respect of her staff
post situation	Mary	hasInPossession	the respect of her staff

"Ford secured the European market."

pre situation	Ford	notHasInPossession	the European market
post situation	Ford	hasInPossession	the European market

-Giving subclassOf: ChangeOfPossession

The subclass of ChangeOfPossession where a person gives something to someone else.

Class mappings:
closeMatch: fn:Sending
closeMatch: fn:Giving
closeMatch: fn:Supply
closeMatch: sumo:Giving

For the assertions and role mappings, see: ChangeOfPossession.

EXAMPLES:

"Mary gave John a nice bouquet."

pre situation	Mary	hasInPossession	a nice bouquet
	John	notHasInPossession	a nice bouquet
post situation	Mary	notHasInPossession	a nice bouquet
	John	hasInPossession	a nice bouquet

"The US shipped tents and food to Indonesia after the tsunami."

pre situation	the US	hasInPossession	tents and food
	Indonesia	notHasInPossession	tents and food
post situation	the US	notHasInPossession	tents and food
	Indonesia	hasInPossession	tents and food

-HavingAValue subclassOf: StaticEvent

"The subclass of StaticEvent where something is having some value."

Class mappings:
closeMatch: fn:Amounting_to.

Role mappings:
value-attribute: fn:Attribute
value: fn:Value

Assertions:

during situation	value-attribute	hasValue	value

EXAMPLE:

"Maries income amounted to 100.000 euro a year."

during situation	Maries income	hasValue	100.000 euro

-HavingInPossession subclassOf: StaticEvent

"Static event where someone has something in possession."

Class mappings:
closeMatch: fn:Possession
closeMatch: fn:Retaining

Role mappings:
possession-owner: fn:Agent, fn:Owner
possession-theme: fn:Theme, fn:Goods, fn:Possession

Assertions:

during situation	possession-owner	hasInPossession	possession-theme

EXAMPLES:

"Tata Steel has 10.000 employees."

during situation	Tata Steel	hasInPossession	10.000 employees

"Mary owns a house in Spain."

during situation	Mary	hasInPossession	a house in Spain

"The US retains political support from Europe."

during situation	The US	hasInPossession	political support from Europe

"Mary kept her old wedding gown."

during situation	Mary	hasInPossession	her old wedding gown

-Importing: subclassOf: Buying
"The subclass of Buying where goods are imported from some country in exchange for money."

Class mappings:
closeMatch: fn:Importing
relatedMatch: sumo:Exporting

For assertions and role mappings, see: FinancialTransaction.

EXAMPLES:

"Canada imported 45.000 cars from Europe last year."

pre situation	Europe	hasInPossession	45.000 cars
	Canada	notHasInPossession	45.000 cars
post situation	Europe	notHasInPossession	45.000 cars
	Canada	hasInPossession	45.000 cars

"Iran's import of nuclear material was monitored."

pre situation	Iran	notHasInPossession	nuclear material
post situation	Iran	hasInPossession	nuclear material

-Increasing subclassOf: QuantityChange
"The subclass of InternalChange where some physical quantity or value is increased."

Class mappings:
broadMatch: fn:Change_of_quantity_of_possession
broadMatch: fn:Cause_change_of_position_on_a_scale
broadMatch: fn:Change_position_on_a_scale
broadMatch: fn:Proliferating_in_number
broadMatch: fn: Expansion
broadMatch: fn: Cause_expansion
closeMatch: fn:Cause_proliferation_in_number
closeMatch: sumo:Increasing

Role mappings:
quantity-item: fn: Item, fn:Possession, fn:Set
quantity-attribute: fn:Attribute, fn:Dimension
quantity-ratio: fn:Size_change, fn:Difference
quantity-value_1: fn:Initial_value, fn:Initial_number, fn:Initial_size, fn:Value_1
quantity-value_2: fn:Final_value, fn:Final_number, fn:Value_2, fn:Result_size

Assertions:

pre situation	quantity-item	hasAttribute	quantity-attribute
	quantity-attribute	hasRelativeValue	-

	quantity-attribute	hasValue	quantity-value_1
post situation	quantity-item	hasAttribute	quantity-attribute
	quantity-attribute	hasRelativeValue	+
	quantity-attribute	hasValue	quantity-value_2
	quantity-item	hasRelativeIncrease	quantity-ratio

Note that quantity-attribute is modeled with an existential restriction that allows to create a blank node in the named graph.

EXAMPLES:

"Apple raised the price of the Iphone from 500 to 600 dollar."

pre situation	Iphone	hasAttribute	price
	price	hasRelativeValue	-
	price	hasValue	500
post situation	Iphone	hasAttribute	price
	price	hasRelativeValue	+
	price	hasValue	600

"Ford increased the production with 2%."

pre situation	production	hasAttribute	:asd123
	:asd123	hasRelativeValue	-
post situation	production	hasAttribute	:asd123
	:asd123	hasRelativeValue	+
	production	hasRelativeIncrease	2%

"Their debt tripled in nine years."

pre situation	their debt	hasRelativeValue	-
post situation	their debt	hasRelativeValue	+

"He widened his eyes."

pre situation	his eyes	hasAttribute	:zxc234
	:zxc234	hasRelativeValue	-
post situation	his eyes	hasAttribute	:zxc234
	:zxc234	hasRelativeValue	+

"The balloon expanded with 2 centimetres".

pre situation	the balloon	hasAttribute	:abc123
	:abc123	hasRelativeValue	-
post situation	the balloon	hasAttribute	:abc123
	:abc123	hasRelativeValue	+
	the balloon	hasRelativeIncrease	2 centimetres

-Injuring subclassOf: Damaging
"The subclass of Damaging where someone gets injured (mentally and/or physically)."

<u>Class mappings:</u>
closeMatch: fn:Cause_harm
closeMatch: fn:Experience_bodily_harm
closeMatch: sumo:Injuring

For the assertions and role mappings, see: Damaging.

EXAMPLES:

"Marie wounded John."

pre situation	John	inState	:qwe556
	qwe556	hasRelativeValue	+
post situation	John	inState	:zxc678
	:zxc678	hasRelativeValue	-
post situation:	John	isDamaged	true

"John broke his leg after falling off the stage"

pre situation	John, his leg	inState	:abc123
	:abc123	hasRelativeValue	+
post situation	John, his leg	inState	:abc124
	:abc124	hasRelativeValue	-
post situation:	John, his leg	isDamaged	true

"Mary broke his leg with her bare hands!"

pre situation	his leg	inState	:jkl234
	:jkl234	hasRelativeValue	+
post situation	his leg	inState	:asd345
	:asd345	hasRelativeValue	-
post situation:	his leg	isDamaged	true

-Installing subclassOf: Placing

"The subclass of Placing where some entity is put in a new and fixed location, e.g. the installation of fixtures."

<u>Class mappings:</u>
closeMatch: fn:Installing
closeMatch: sumo:Installing

For the assertions and role mappings, see: Translocation.

EXAMPLES:

"Mary installed a new engine in her Land Rover Defender."

pre situation	a new engine	notAtPlace	Land Rover Defender
post situation	a new engine	atPlace	Land Rover Defender

"John confirmed the installation of cameras in the offices."

pre situation	cameras	notAtPlace	in the offices
post situation	cameras	atPlace	in the offices

-IntentionalEvent subclassOf:DynamicEvent

"The subclass of DynamicEvent where some event is carried out by some cognitive agent(s) and with some specific purpose."

<u>Class mappings:</u>
closeMatch: fn:Intentionally_act
sumo: IntentionalProcess

No assertions are defined for this class.

-InternalChange subclassOf: DynamicEvent

"The subclass of DynamicEvent where some internal quality of an item changes."

Class mappings:
closeMatch: sumo:InternalChange

No assertions are defined for this class.

-Investing subclassOf: FinancialTransaction

The subclass ofFinancialTransaction where a person or company invests some asset in either another or its own company with the prospect of some future profit.

Class mappings:
closeMatch: sumo:Investing

For the assertions, see: FinancialTransaction.

-JoiningAnOrganization subclassOf: IntentionalEvent

"The subclass of IntentionalEvent where someone starts working as an employee for some organization."

Class mappings:
closeMatch: fn:Hiring,
closeMatch: fn:Get_a_job
broadMatch: sumo:JoiningAnOrganization

Role mappings:
employment-employee: fn:Employee
employment-employer: fn:Employer
employment-function: fn:Position
employment-value: fn:Compensation
employment-task: fn:Task
employment-attribute: -

Assertions:

pre situation	employment-employee	notEmployedAt	employment-employer
post situation	employment-employee	employedAt	employment-employer
	employment-employee	isEmployed	true
	employment-employee	hasFunction	employment-function
	employment-employee	hasTask	employment-task
	employment-employee	hasAttribute	employment-attribute
	employment-attribute	hasValue	employment-value

Note that employment-attribute is modeled with an existential restriction that allows to create a blank node in the named graph.

EXAMPLES:

"Ford hired Mary as their new CEO for 100.000 euro."

pre situation	Mary	notEmployedAt	Ford
post situation	Mary	isEmployed	true
	Mary	employedAt	Ford
	Mary	hasFunction	new CEO
	Mary	hasAttribute	:abc124
	:abc124	hasValue	100.000 euro

"John was hired to clean the house."

pre situation	-		
post situation	John	isEmployed	true
	John	hasTask	to clean the house

"John signed on with Marie to clean her house."

pre situation	John	notEmployedAt	Marie
post situation	John	isEmployed	true
	John	employedAt	Marie
	John	hasTask	to clean her house

-Killing subclassOf: Destroying

"The subclass of Destroying where animate beings are killed."

Class mappings:
closeMatch: fn:Execution
closeMatch: fn:Killing
closeMatch: sumo:Killing

For assertions and role mappings, see: Destroying.

EXAMPLES:

"Mary was executed by three men in black ties."

pre situation	Mary	exist	true
post situation	Mary	exist	false

"Low levels of oxygen asphyxiated the fish in John's pond."

pre situation	the fish in John's pond	exist	true
post situation	the fish in John's pond	exist	false

-Leaving subclassOf:Translocation

"The subclass of Translocation where someone or something leaves a location."

Class mappings:
closeMatch: fn:Vehicle_departure_initial_state
closeMatch: fn:Departing
closeMatch: fn:Setting_out
closeMatch: fn:Quitting_a_place
closeMatch: sumo:Leaving.

For the assertions and role mappings, see: Translocation.

EXAMPLES:

"John set out from Lake Louise in a canoe."

pre situation	John	atPlace	Lake Louise
post situation	John	notAtPlace	Lake Louise

"John left for Lake Michigan."

pre situation	John	notAtPlace	Lake Michigan
post situation	John	atPlace	Lake Michigan*

**Note that Johns arrival at Lake Michigan is not certain.*

-LeavingAnOrganization subclassOf: IntentionalEvent

"The subclass of IntentionalEvent where a person stops working as an employee for an organization."

Class mappings:
closeMatch: fn:Quitting,
closeMatch: fn:Firing
closeMatch: sumo:TerminatingEmployment

Role mappings:
employment-employee: fn:Employee
employment-employer: fn:Employer
employment-function: fn:Position
employment-task: fn:Task

Assertions:

pre situation	employment-employee	employedAt	employment-employer
	employment-employee	isEmployed	true
	employment-employee	hasFunction	employment-function
	employment-employee	hasTask	employment-task
post situation	employment-employee	notEmployedAt	employment-employer

EXAMPLES:

"Ford fired Mary as their CEO."

pre situation	Mary	employedAt	Ford
	Mary	isEmployed	true
	Mary	hasFunction	CEO
post situation	Mary	notEmployedAt	Ford

"John was fired from cleaning the house."

pre situation	John	isEmployed	true
	John	hasTask	cleaning the house
post situation	-		

"John left Ford."

pre situation	John	employedAt	Ford
post situation	John	notEmployedAt	Ford

-Lending subclassOf:Giving

“The subclass of Giving where a person gives something in possession for some period of time after which the item should be given back.”

Class mappings:
closeMatch: fn:Lending
closeMatch: sumo:Lending

For the assertions and role mappings, see: ChangeOfPossession.

EXAMPLE:

"Mary loaned her car to John."

pre situation	Mary	hasInPossession	her car
	John	notHasInPossession	her car
post situation	Mary	notHasInPossession	her car
	John	hasInPossession	her car

-Meeting subclassOf: StaticEvent

"The static event class where people meet each other, usually intentional and for some purpose."

Class mappings:
closeMatch: fn:Come_together
closeMatch: fn:Assemble
closeMatch: fn:Social_event
closeMatch: sumo:Meeting

Role mappings:
meeting-participant: Party_1, Party_2, fn:Attendee, fn:Host, fn:Individuals, fn:Group, fn:Configuration
meeting-place: fn:Place

Assertions:

during situation	meeting-participantatPlace	meeting-place
	meeting-participantinMeeting	true

EXAMPLES:

"The Republicans convened in New York to discuss the program."

during situation	the Republicans	atPlace	New York
	the Republicans	inMeeting	true

"John meets Marie in New York"

during situation	John	atPlace	New York
	Marie	atPlace	New York
	John, Marie	inMeeting	true

"The whole group attended the party"

during situation	the whole group	inMeeting	true

-Merging subclassOf: InternalChange

"The subclass of InternalChange where two entities are merged into a whole."

Class mappings:
closeMatch: fn:Amalgamation
closeMatch: fn:Cause_to_amalgamate
closeMatch: sumo:Combining

Role mappings:
merging-theme_1: fn:Part_1, fn:Parts
merging-theme_2: fn:Part_2
merging-theme_3: fn:Whole

Assertions:

pre situation	merging-theme_1	exist	true
	merging-theme_2	exist	true
	merging-theme_3	exist	false

post situation:	merging-theme_1	exist	false
	merging-theme_2	exist	false
	merging-theme_3	exist	true

EXAMPLES:

"In 1980, EBC merged with KPN into KPN-BC."

pre situation	EBC	exist	true
	KPN	exist	true
	KPN-BC	exist	false
post situation	EBC	exist	false
	KPN	exist	false
	KPN-BC	exist	true

"John blended the herbs and the eggs."

pre situation	the herbs and the eggs	exist	true
post situation	the herbs and the eggs	exist	false

-Motion subclassOf: DynamicEvent

"The subclass of DynamicEvent where some entity moves."

Class mappings:
closeMatch: fn:Motion
closeMatch: sumo:Motion

No assertions are defined for this class.

-Paying subclassOf: FinancialTransaction

"The subclass of FinancialTransaction where some financial asset is given in exchange for some item or in discharge of a debt."

Class mappings:
closeMatch: fn:Commerce_pay

For the assertions and role mappings, see: FinancialTransaction.

EXAMPLES:

"Ford paid Chrysler 40.000 dollar for John's idea."

pre situation	Ford	notHasInPossession	John's idea
	Chrysler	hasInPossession	John's idea
	Ford	hasInPossession	40.000 dollar
	Chrysler	notHasInPossession	40.000 dollar
post situation	Ford	hasInPossession	John's idea
	Chrysler	notHasInPossession	John's idea
	Ford	notHasInPossession	40.000 dollar
	Chrysler	hasInPossession	40.000 dollar
during situation	John's idea	hasValue	40.000 dollar

"Mary paid the bill."

pre situation	Mary	hasInPossession	the bill

post situation	Mary	notHasInPossession	the bill

-Placing subclassOf:Translocation

"The subclass of Translocation where some entity is put in a new location."

Class mappings:
closeMatch: fn:Placing
closeMatch: sumo:Putting

For the assertions and role mappings, see: Translocation.

EXAMPLES:

"While thinking of Mary, John put the flowers in a vase."

pre situation	flowers	notAtPlace	in a vase
post situation	flowers	atPlace	in a vase

"Mary loaded all her belongings in the car."

pre situation	her belongings	notAtPlace	in the car
post situation	her belongings	atPlace	in the car

"The sea deposited dead fish on the beach."

pre situation	dead fish	notAtPlace	on the beach
post situation	dead fish	atPlace	on the beach

-QuantityChange subclassOf: InternalChange

"The subclass of InternalChange where some quantity is altered."

Class mappings:
closeMatch: sumo: QuantityChange

No assertions are defined for this class.

-Removing subclassOf: Translocation

"The subclass of Translocation where some entity is taken away from its location."

Class mappings:
closeMatch: fn:Removing
closeMatch: sumo:Removing

For the assertions and role mappings, see: Translocation.

EXAMPLES:

"John removed all the evidence from the archive."

pre situation	the evidence	atPlace	the archive
post situation	the evidence	notAtPlace	the archive

"Mary evacuated the employees from the burning factory."

pre situation	the employees	atPlace	the burning factory
post situation	the employees	notAtPlace	the burning factory

"The Maserati was unloaded from the Boeing 747."

pre situation	the Maserati	atPlace	the Boeing 747
post situation	the Maserati	notAtPlace	the Boeing 747

"John removed all his books."

pre situation	-
post situation	-

-Renting subclassOf: Getting

"The subclass of Getting where a person gets something in possession from someone else for some period in exchange for money."

<u>Class mappings:</u>
closeMatch: fn:Renting
closeMatch: sumo:Renting

For the assertions and role mappings, see: ChangeOfPossession.

EXAMPLES:

"John leased his Peugeot from ELB."

pre situation	John	notHasInPossession	his Peugeot
	ELB	hasInPossession	his Peugeot
post situation	John	hasInPossession	his Peugeot
	ELB	notHasInPossession	his Peugeot

"Mary rented a room from an old lady."

pre situation	Mary	notHasInPossession	a room
	an old lady	hasInPossession	a room
post situation	Mary	hasInPossession	a room
	an old lady	notHasInPossession	a room

-RentingOut subclassOf: Giving

"The subclass of Giving where a person gives something in possession for some period in exchange for money."

<u>Class mappings:</u>
closeMatch: fn:Renting_out

For the assertions and role mappings, see: ChangeOfPossession.

EXAMPLES:

"The old lady rented a room to Mary."

pre situation	Mary	notHasInPossession	a room
	an old lady	hasInPossession	a room
post situation	Mary	hasInPossession	a room
	an old lady	notHasInPossession	a room

"Mary rented the garage out."

pre situation	Mary	hasInPossession	the garage
post situation	Mary	notHasInPossession	the garage

-Replacing subclassOf: IntentionalEvent

“The subclass of IntentionalEvent were someone or something is replaced with someone or something else in a specific role or function."

Class mappings:
closeMatch: fn:Replacing
closeMatch: fn: Take_place_of
closeMatch: fn:Change_of_leadership
closeMatch: sumo:Substituting

Role mappings:
replacing-entity_1: fn:Old, fn:Old_order, fn:Old_leader
replacing-entity_2: fn:New, fn:New_leader
replacing-entity_3: fn:Agent
replacing-function: fn:Role, fn:Function

Assertions:

pre situation	replacing-entity_1	hasFunction	replacing-function
	replacing-entity_2	notHasFunction	replacing-function
	replacing-entity_1	inFunctionFor	replacing-entity_3
	replacing-entity_1	inFunction	true
	replacing-entity_2	inFunction	false
post situation	replacing-entity_1	notHasFunction	replacing-function
	replacing-entity_2	hasFunction	replacing-function
	replacing-entity_2	inFunctionFor	replacing-entity_3
	replacing-entity_1	inFunction	false
	replacing-entity_2	inFunction	true

EXAMPLES:

"Peter replaced Mary by John as CEO of Apple."

pre situation	Mary	hasFunction	CEO of Apple
	John	notHasFunction	CEO of Apple
	Mary	inFunctionFor	Peter
	Mary	inFunction	true
	John	inFunction	false
post situation	Mary	notHasFunction	CEO of Apple
	John	hasFunction	CEO of Apple
	John	inFunctionFor	Peter
	Mary	inFunction	false
	John	inFunction	true

"Mary replaced her Ford Taunus for a Peugeot 205."

pre situation	Ford Taunus	inFunctionFor	Mary
	Ford Taunus	inFunction	true
	Renault 205	inFunction	false
post situation	Peugeot 205	inFunctionFor	Mary
	Ford Taunus	inFunction	false
	Peugeot 205	inFunction	true

"Vinyl was replaced by the compact disc in the early eighties."

pre situation	vinyl	inFunction	true
	compact disc	inFunction	false
post situation	compact disc	inFunction	true

	vinyl	inFunction	false

"Amsterdam installed Mary as the new mayor."

pre situation	Mary	notHasFunction	mayor
	Mary	inFunction	false
post situation	Mary	hasFunction	mayor
	Mary	inFunctionFor	Amsterdam
	Mary	inFunction	true

"The rebellion against the Lannisters."

pre situation	Lannisters	inFunction	true
post situation	Lannisters	inFunction	false*

**Note that, due to the lexical units associated to a FrameNet frame, the triggered assertions can be too strong.*

-Selling subclassOf: FinancialTransaction

``The subclass of FinancialTransaction where some entity changes of ownership in exchange for money."

<u>Class mappings:</u>
closeMatch: fn:Commerce_sell
closeMatch: sumo:Selling

For the assertions and role mappings, see: FinancialTransaction.

EXAMPLES:

"In 2013, Ford sold 10.000 cars."

pre situation	Ford	hasInPossession	10.000 cars
post situation	Ford	notHasInPossession	10.000 cars

"The Catholic church auctioned off 20 churches to project developers."

pre situation	Catholic church	hasInPossession	20 churches
	project developers	notHasInPossession	20 churches
post situation	Catholic church	notHasInPossession	20 churches
	project developers	hasInPossession	20 churches

"Mary sold the plot of land to John for 10.000 dollar."

pre situation	Mary	hasInPossession	the plot of land
	John	notHasInPossession	the plot of land
	Mary	notHasInPossession	10.000 dollar
	John	hasInPossession	10.000 dollar
post situation	Mary	notHasInPossession	the plot of land
	John	hasInPossession	the plot of land
	Mary	hasInPossession	10.000 dollar
	John	notHasInPossession	10.000 dollar
during situation	the plot of land	hasValue	10.000 dollar

-Separating subclassOf: InternalChange

"The subclass of InternalChange where some whole is split into parts."

<u>Class mappings:</u>
closeMatch: fn:Becoming_separated

closeMatch: fn:Separating
closeMatch: sumo:Separating

<u>Role mappings:</u>
separating-theme_1: fn:Part_1, fn:Parts
separating-theme_2: fn:Part_2
separating-theme_3: fn:Whole

<u>Assertions:</u>

pre situation	separating-theme_1	exist	false
	separating-theme_2	exist	false
	separating-theme_3	exist	true
post situation	separating-theme_1	exist	true
	separating-theme_2	exist	true
	separating-theme_3	exist	false

EXAMPLES:

"The machine split the water into hydrogen and oxygen."

pre situation	hydrogen and oxygen	exist	false
	water	exist	true
post situation	hydrogen and oxygen	exist	true
	water	exist	false

"Mary divided the pile of cutlery into groups of six."

pre situation	groups of sixexist	false	
	pile of cutlery	exist	true
post situation	groups of sixexist	true	
	pile of cutlery	exist	false

"The auctioneer separated the hatchbacks from the saloons.*”

pre situation	the hatchbacks	exist	false
	the saloons	exist	false
post situation	the hatchbacks	exist	true
	the hatchbacks	exist	true

**Note that separating-theme_3 (the whole collection of cars) remains implicit in this example.*

"The partition of Germany in 1945."

pre situation	Germany	exist	true
post situation	Germany	exist	false

-StartingAnActivity subclassOf: IntentionalEvent
“The subclass of IntentionalProcess where someone intentionally starts an activity."

<u>Class mappings:</u>
closeMatch: fn:Activity_start

<u>Role mappings:</u>
activity: fn:Activity
activity-agent: fn:Agent

<u>Assertions:</u>

pre situation	activity	exist	false
post situation	activity	exist	true
	activity-agent	involvedIn	activity

"Ford started the production of the Taunus in 1979."

pre situation	production of the Taunus	exist	false
post situation	production of the Taunus	exist	true
	Ford	involvedIn	production of the Taunus

"The government began protecting the peat bogs in Ost-Friesland."

pre situation	protecting the peat bogs in Ost-Friesland	exist	false
post situation	protecting the peat bogs in Ost-Friesland	exist	true
	the government	involvedIn	protecting the peat bogs in Ost-Friesland.

-StaticEvent StaticEvent is the top node of the static event class hierarchy.
"A StaticEvent is an entity which is associated with a period of time
where a set of propositions is true."

Class mappings:
closeMatch: fn:State

No assertions are defined for this class.

-Stealing subclassOf: Taking
"The subclass of Taking where a person takes something without permission of the owner."

Class mappings:
closeMatch: fn:Theft
closeMatch: sumo:Stealing

For the assertions and class mappings, see: ChangeOfPossession.

EXAMPLES:

"John shoplifted a sweater from the department store."

pre situation	department store	hasInPossession	sweater
	John	notHasInPossession	sweater
post situation	department store	notHasInPossession	sweater
	John	hasInPossession	sweater

"Marie stole a sweater from John."

pre situation	John	hasInPossession	a sweater
	Marie	notHasInPossession	a sweater
post situation	John	notHasInPossession	a sweater
	Marie	hasInPossession	a sweater

"Massive theft of documents from the Stasi archives."

pre situation	Stasi archives	hasInPossession	documents
post situation	Stasi archives	notHasInPossession	documents

-StoppingAnActivity subclassOf:IntentionalEvent
"The subclass of IntentionalProcess where some agent intentionally stops an activity."

Class mappings:
closeMatch: fn:Activity_stop

Role mappings:

activity: fn:Activity
activity-agent: fn:Agent

<u>Assertions:</u>

pre situation	activity	exist	true
	activity-agent	involvedIn	activity
post-situation	activity	exist	false
	activity-agent	notInvolvedIn	activity

"Ford terminated the negotiations with Peugeot."

pre situation	negotiations with Peugeot	exist	true
	Ford	involvedIn	negotiations with Peugeot
post situation	negotiations with Peugeot	exist	false
	Ford	notInvolvedIn	negotiations with Peugeot

"John's treatment was discontinued."

pre situation	John's treatment	exist	true
post situation	John's treatment	exist	false

-Taking subclassOf: Getting

"The subclass of Getting where a person takes something without giving something in return."

<u>Class mappings</u>:
closeMatch: fn:Taking
closeMatch: sumo:UnilateralGetting

For the assertions and role mappings, see: ChangeOfPossession

EXAMPLES:

"The police seized financial documents from the private equity fund."

pre situation	the police	notHasInPossession	financial documents
	private equity fund	hasInPossession	financial documents
post situation	the police	hasInPossession	financial documents
	private equity fund	notHAsInPossession	financial documents

"Mary took a beer from the refrigerator."

pre situation	Mary	notHasInPossession	a beer
	the refrigerator	hasInPossession	a beer
post situation	Mary	hasInPossession	a beer
	the refrigerator	notHasInPossession	a beer

-Translocation subclassOf:Motion

"The subclass of Motion where physical objects or animate beings change from location."

<u>Class mappings:</u>
closeMatch: fn:Self_motion
closeMatch: fn:Cotheme
closeMatch: fn:Traversing
closeMatch: fn:Use_vehicle
closeMatch: fn:Intentional_traversing
closeMatch: fn:Ride_vehicle

closeMatch: fn:Travel
closeMatch: fn:Operate_vehicle
closeMatch: fn:Cause_motion
closeMatch: sumo:Translocation

Role mappings:
translocation-theme: fn:Self_mover, fn: Theme, fn:Driver, fn:Traveler, fn:Vehicle, fn:Escapee, fn:Cotheme, fn:Component, fn:Individuals.
translocation-source: fn:Source, fn: Undesirable_location
translocation-goal: fn:Goal, fn: Intended_goal, fn: Goal_area

Assertions:

pre situation:	translocation-theme	atPlace	translocation-source
	translocation-theme	notAtPlace	translocation-goal
post situation:	translocation-theme	atPlace	translocation-goal
	translocation-theme	notAtPlace	translocation-source

EXAMPLE:

"John drove from New York to Atlanta."

pre situation	John	atPlace	New York
	John	notAtPlace	Atlanta
post situation	John	atPlace	Atlanta
	John	notAtPlace	New York

-Transportation subclassOf:Transportation

"The subclass of Translocation where physical objects and animate beings together change from location and the physical object is not the means of translocation."

Class mappings:
closeMatch: fn:Bringing
closeMatch: fn:Delivery
closeMatch: sumo:Transportation

For the assertions and role mappings, see: Translocation

EXAMPLES:

"Mary brought her classic car from the US to England."

pre situation	her classic car	atPlace	US
	her classic car	notAtPlace	England
post situation	her classic car	atPlace	England
	her classic car	notAtPlace	US

"John flew Mary to the nearest hospital."

pre situation	Mary	notAtPlace	hospital
post situation	Mary	atPlace	hospital

"Russian gas deliveries to Europe."

pre situation	gas	atPlace	Russia
	gas	botAtPlace	Russia
post situation	gas	notAtPlace	Russia
	gas	atPlace	Europe

"The postman delivered a letter to Mary's mailbox."

pre situation	a letter	notAtPlace	Mary's mailbox
post situation	a letter	atPlace	Mary's mailbox

"The postman delivered a letter to Mary.*"

pre situation	-
post situation	-

**Note that 'Mary' is a 'Beneficiary' according to FrameNet. The fn:Beneficiary is not mapped to ESO translocation-goal.*

-Working subclassOf: StaticEvent
"Static event where someone is doing work. "

Class mappings:
closeMatch: fn:Working_a_post
closeMatch: fn:Work

Role mappings:
working-entity: fn:Agent

Assertions:

during situation	working-entity	works	true

EXAMPLES:

"John works hard on a new book."

during situation	John	works	true

"John and Mary manned the front desk."

during situation	John and Mary	works	true