This is an exercise from the Merode course at KU Leuven. The exercise is used as an optional take-home assignment.
The solution is given as an EDG (part of a Merode model)
Some of the mistakes from student solutions are specific to Merode, others are universal

Alpha Insurance Company

Requirements Analysis Table

- Helpful tool to create the EDG
- Several solutions are possible
- → not graded

	Enterprise Layer	Enterprise Layer Event	Information System Service	Business Process Layer
After the customer's profile analysis, a preliminary offer on an insurance product is made to the customer either in person or by email.	Customer, (offer), insurance product(policy),	Make offer	Send email	Make offer in person or by email
If the customer agrees to the offer, a contract is generated and signed by both parties.	Customer, (offer), contract,	Agree to the offer, sign contract	Register agreement, generate contract, register signing	Sign contract
Afterwards, the client is invoiced (monthly or yearly – depending on the choice made in the contract) according to the price of the insurance product they bought.	Client(Customer), Invoice, contract, insurance product (policy)	-	(Automatical) sending of invoice	Buying insurance product, sending of invoice

Frequently occurring errors

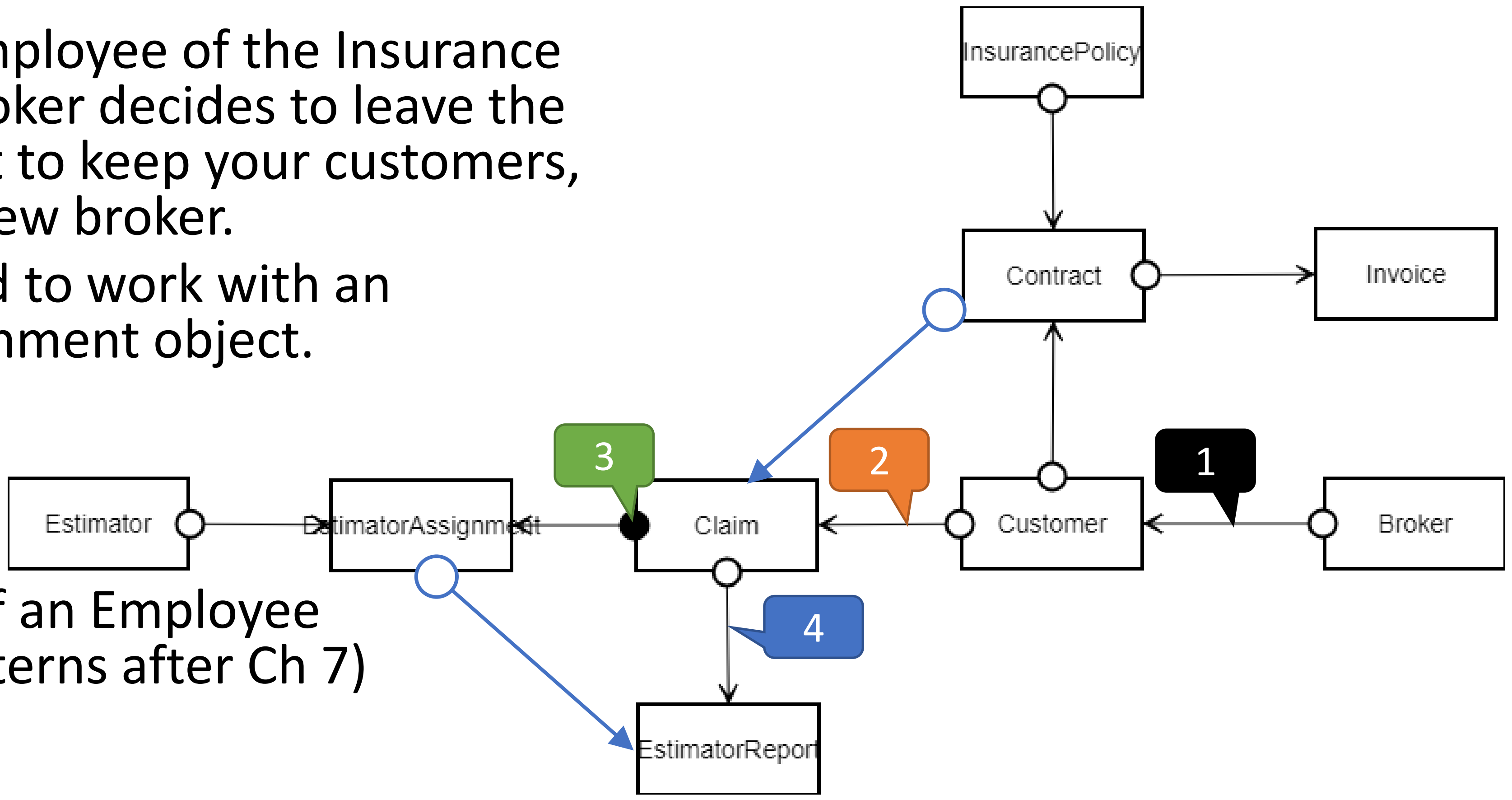
Frequent mistakes

1. Customer is not ED of broker:

   The Broker is an Employee of the Insurance Company. If the Broker decides to leave the company, you want to keep your customers, and just assign a new broker.

   Therefore you need to work with an intermediate Assignment object.

   "Broker" is a role of an Employee (see lecture on patterns after Ch 7)



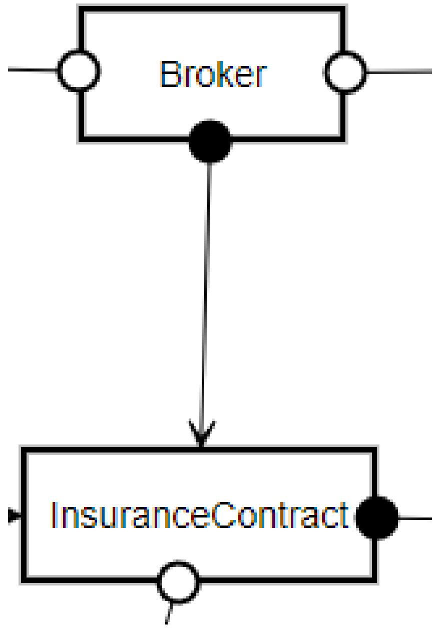
Frequent mistakes

1 (bis) Contract is not ED of broker:

The Broker is an Employee of the Insurance Company. If the Broker decides to leave the company, you want to keep your contracts, and just assign a new broker.

Therefore you need to Contract ED on Customer

Money-generating OT!

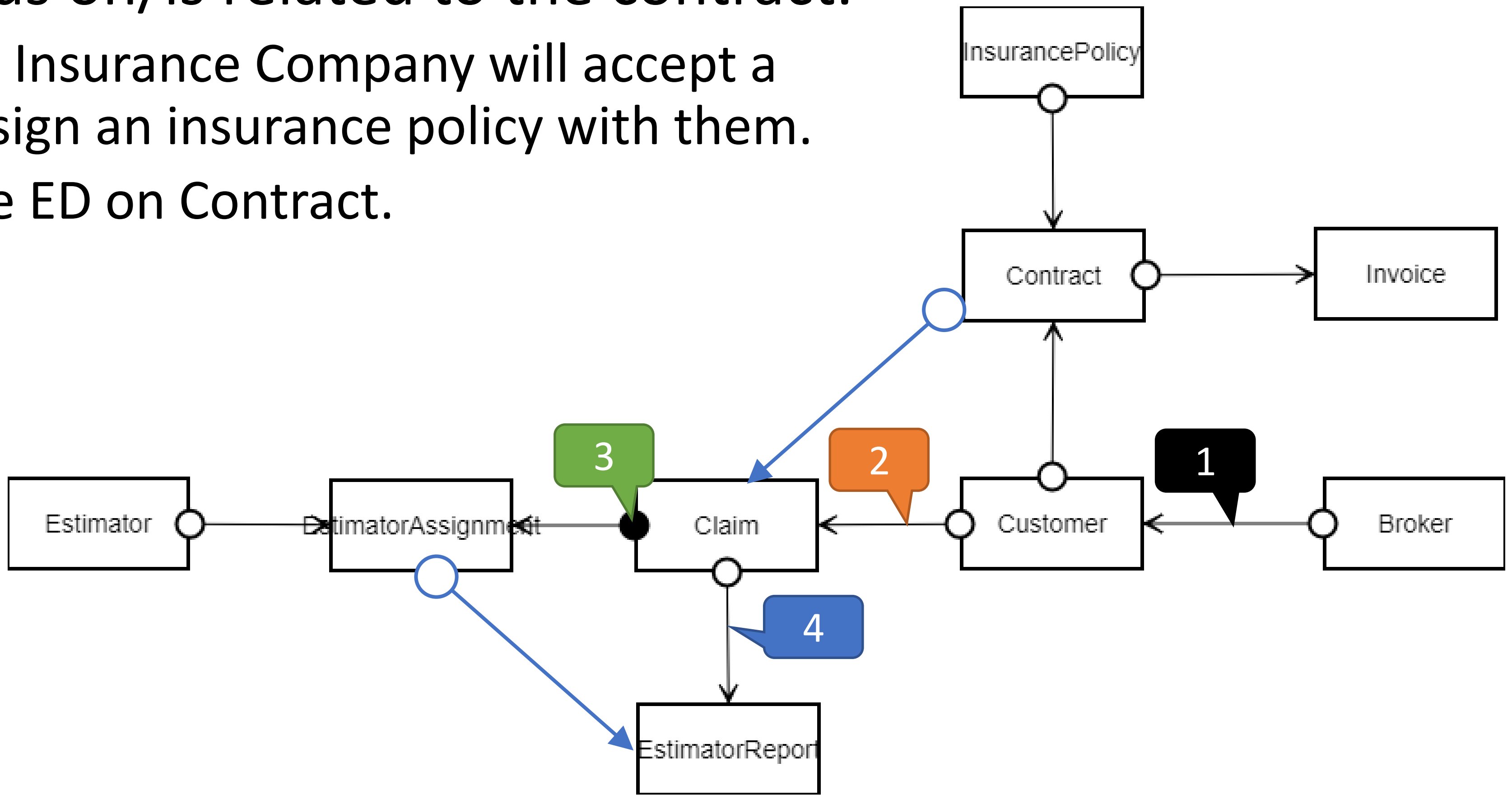


Frequent mistakes

2. The claim depends on/is related to the contract.

   It's unlikely that an Insurance Company will accept a claim if you didn't sign an insurance policy with them.

   So, Claim should be ED on Contract.



Frequent mistakes

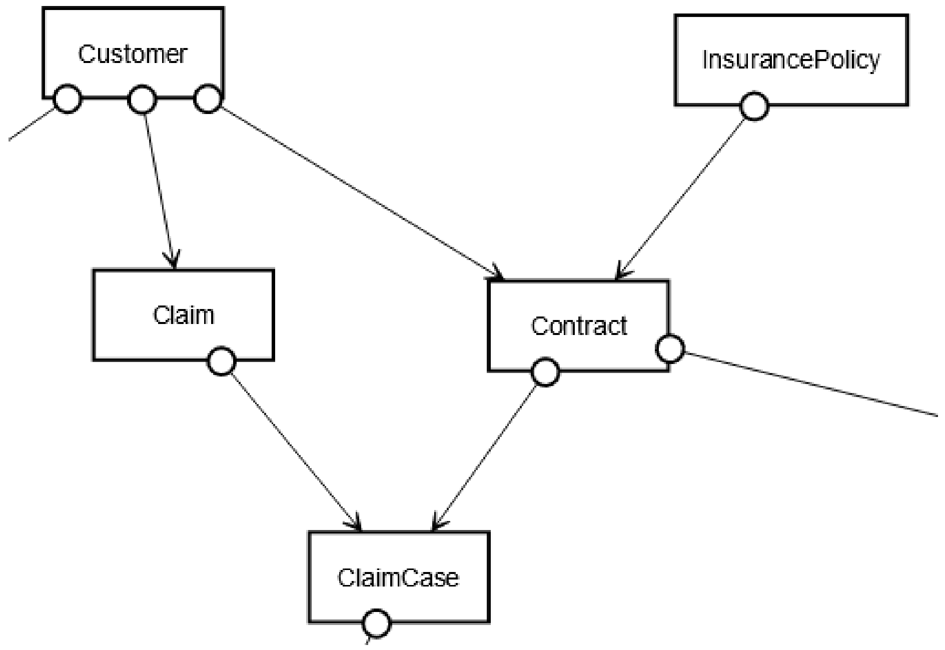
2. The claim depends on/is related to the contract.

It's unlikely that an Insurance Company will accept a claim if you didn't sign an insurance policy with them.

You could make an argument for the claim not to be ED on Contract if you would like a customer to file a claim without having to say in the context of which contract the claim is filed. But in that case, it makes sense to attribute the claim to contract nevertheless. So then you would have a two-step procedure giving rise to a structure with both claims and claim cases.

The claim is what the customer filed

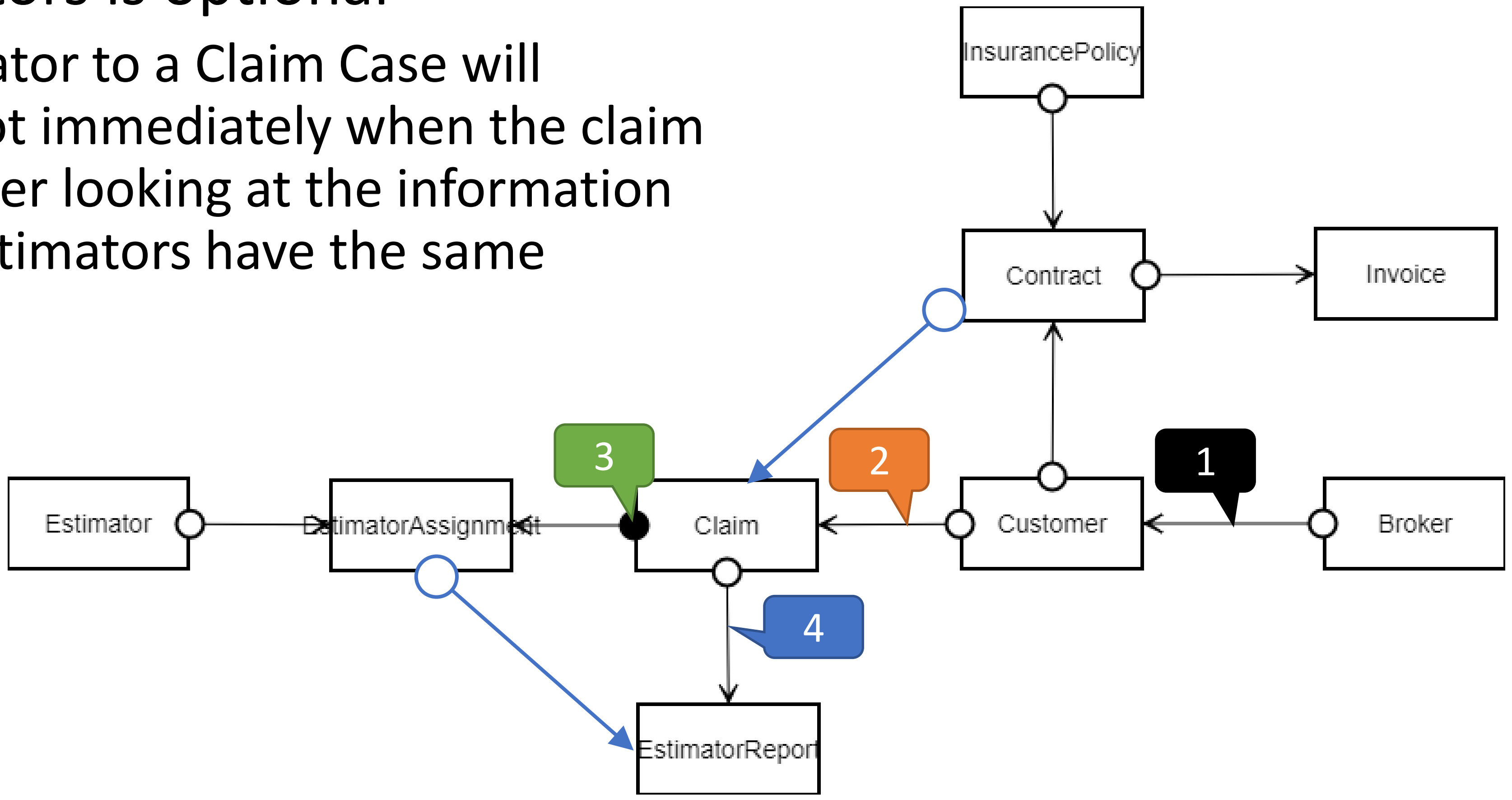
The claimcase is the part of a claim case that is deemed to be covered by a contract.



Frequent mistakes

3. Assigning Estimators is optional

Assigning an Estimator to a Claim Case will typically happen not immediately when the claim is filed, but only after looking at the information as maybe not all estimators have the same qualification



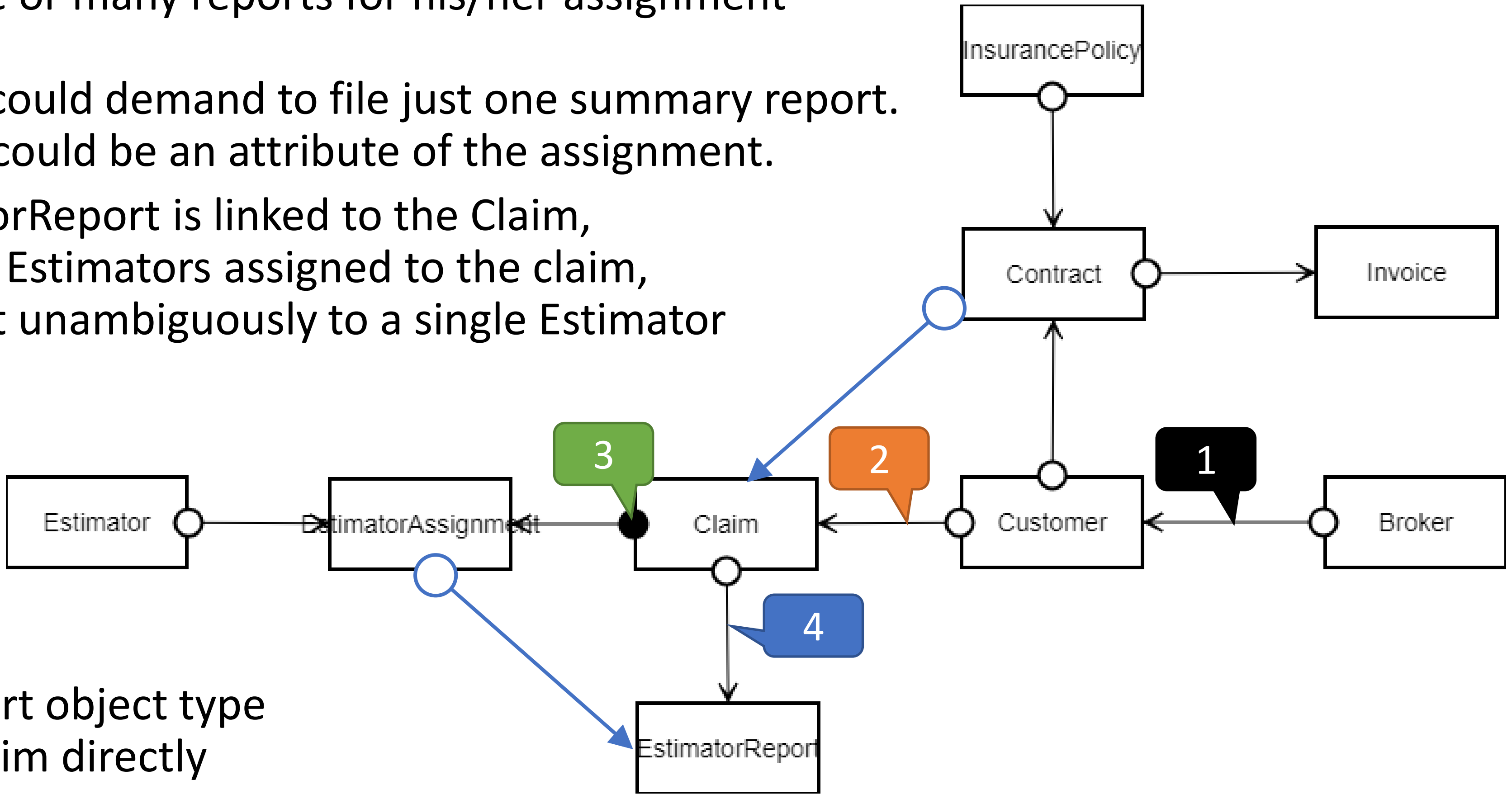
Frequent mistakes

4. The Estimator Report is ED of the assignment

If you decide to have an EstimatorAssignment object type that links an Estimator to a Claim, then the report that results from this assignment should be ED on the assignment object type.

Will the Estimator then file one or many reports for his/her assignment to a specific claim ?
Here we chose many, but one could demand to file just one summary report.
In case of just one report, this could be an attribute of the assignment.

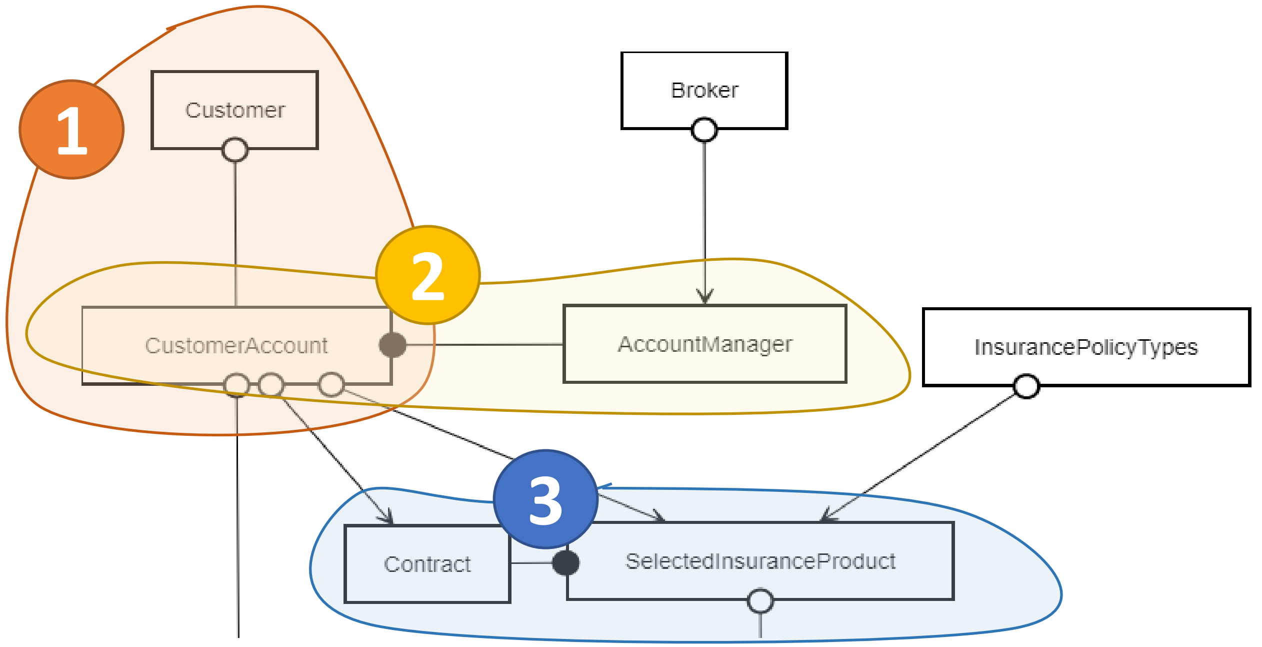
As modelled here, the EstimatorReport is linked to the Claim,
but as there are possibly many Estimators assigned to the claim,
it is impossible to link a Report unambiguously to a single Estimator



It is also valid to have the Report object type linking the Estimator to the Claim directly

Frequent mistakes

- Be careful with mandatory one and optional one associations: Consider
  - using a state inside the LC instead
  - using an attribute instead (cfr. Ch7)

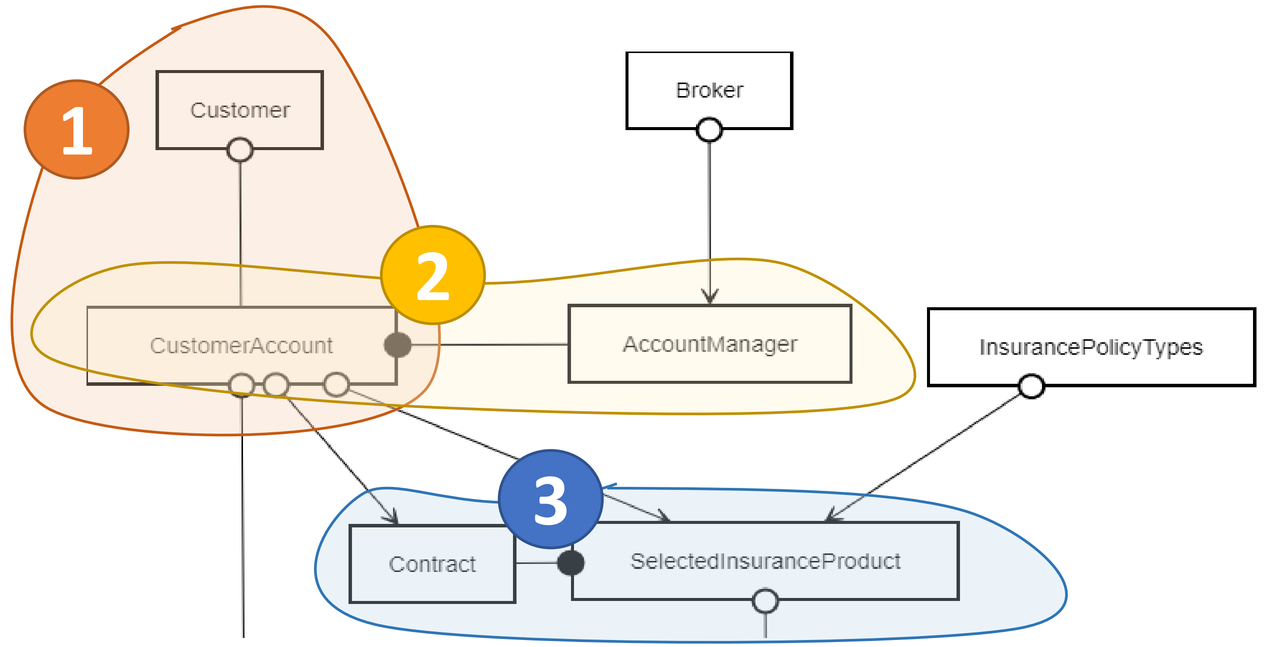


1 If Customer Account is the bank account the customer has, and there is only one, model this as an attribute of Customer.

2 If Customer Account refers to/is the account manager, then Customer Account and Account Manager are the same object type?

Frequent mistakes

- Be careful with mandatory one and optional one associations: Consider
  - using a state inside the LC instead
  - using an attribute instead (cfr. Ch7)

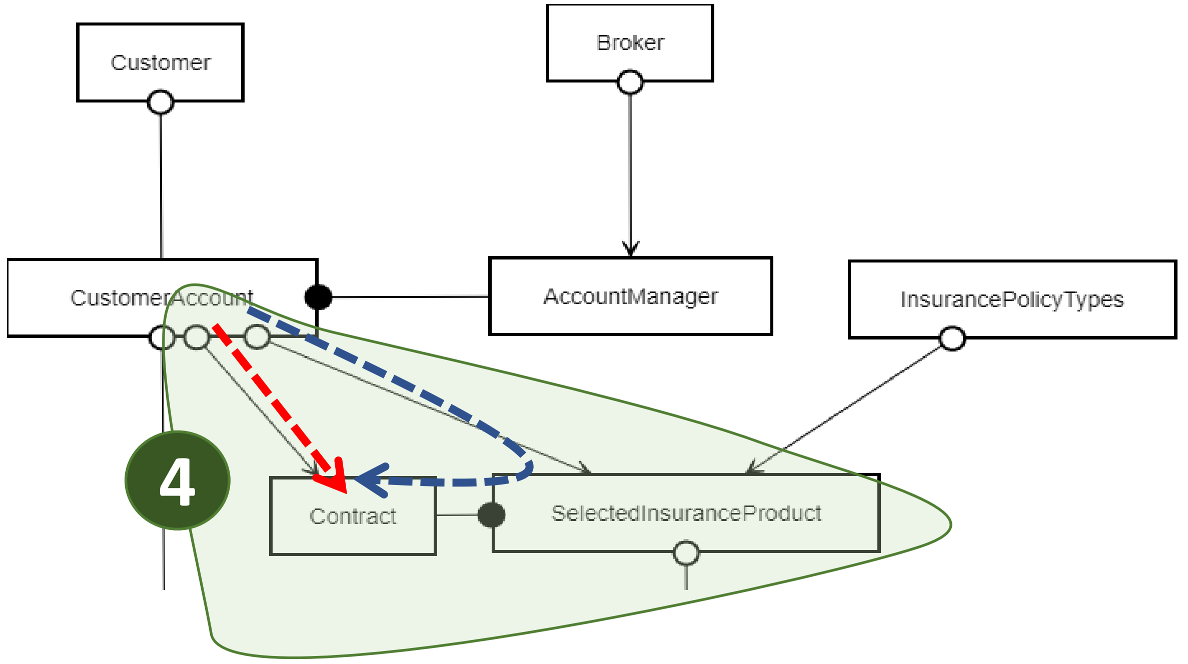


3 The "Selected Insurance Product" or "Offer" could be a state of the "Contract" object.

Given that Contract is ED of "Insurance Policy (Type)", then at the moment of creating a contract you have to select one. So the Contract = SelectedInsuranceProduct ?

Frequent mistakes

- Avoid triangles where one ED relation is the combination of the two other ED relations



4

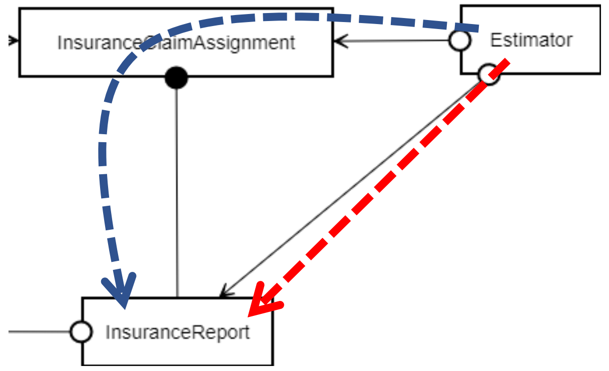
Contract is ED on SelectedInsurance Product

Selected Insurance Product is ED on CustomerAccount

Hence, Contract is indirectly ED on Customer Account.

Thereferore, the RED ED relation can be removed

(Same reasoning for the ED relation between Estimater and Insurance Report)



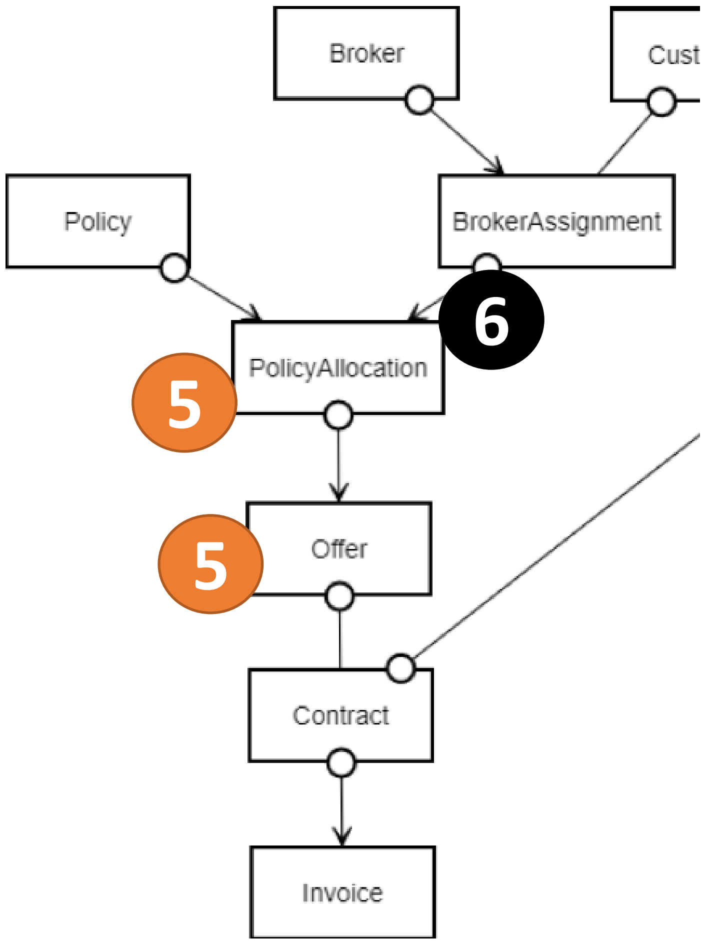
Frequent Mistake

5 Confusing states & Tasks with Object Types

- When the Customer chooses a Policy, this choice may give rise to an object type. That would typically be the "future contract".
- So, having "contract" is enough.
- Sometimes it makes sense to distinguish between "offer" that may or may not give rise to a contract. If you think it's really needed to distinguish between an offer/requested insurance and an actual contract, then carefully motivate why having "Offered" as a state of the "Contract" is not enough.

- Confusing Actor involved in a task with ED

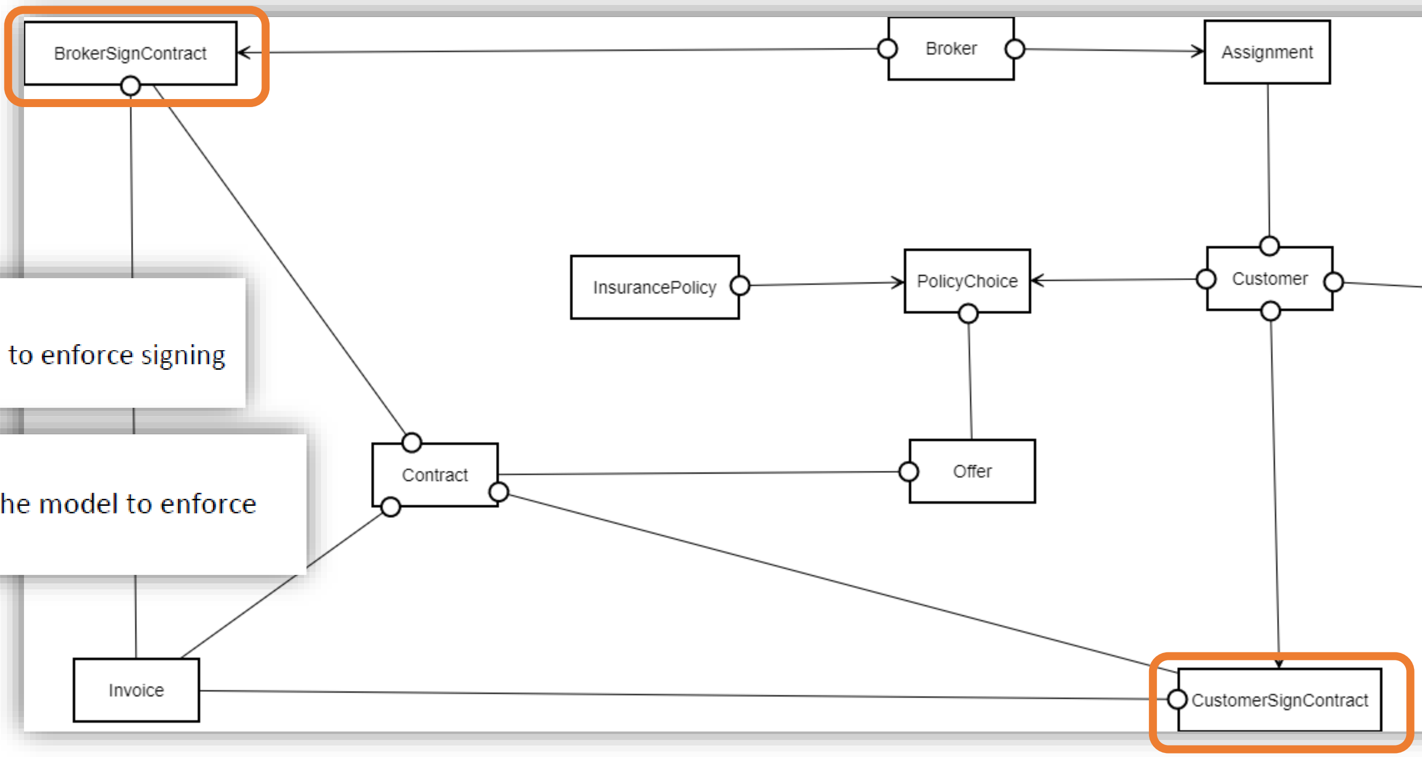
6
  - It's not because the Broker is involved in making the offer, that Offer is ED of Broker(Assignment).



Frequent Mistakes

5 Confusing Tasks and Object types

- The signature of a contract is a task in the BP layer, and may registered by means of an input services that triggers a "sign" event, that causes a state change in the LC of "Contract", moving it from the state "unsigned" to "signed"



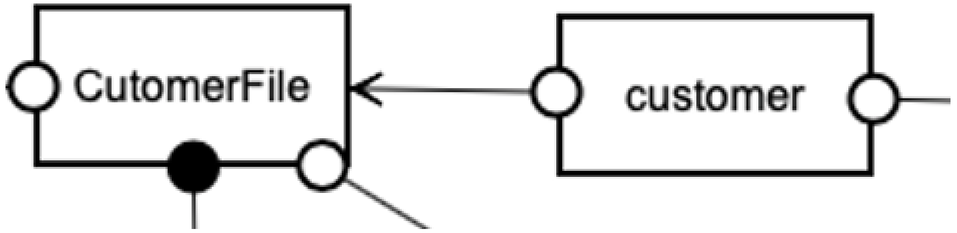
BrokerSignContract
Link between the Broker and Contract. This needs to be added to allow the model to enforce signing the contract by the broker.

CustomerSignContract
Link between the Customer and the Contract. This needs to be added to allow the model to enforce signing the contract by the customer.

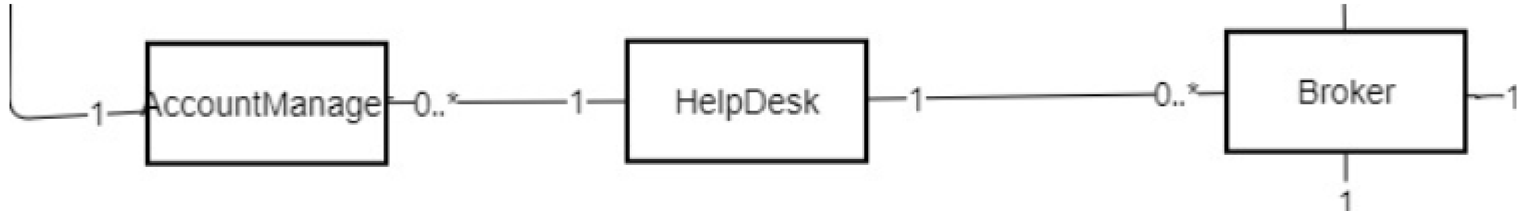
Frequent Mistakes

7 Confusing an Output Service with an Object Type

- The "Customer File" is a report containing all information related to a customer. So this is an output service, not an extra object type.



- Confusing an actor from the Business Process Layer with an Object Type
  - The “Helpdesk” is an actor and is not part of the domain model

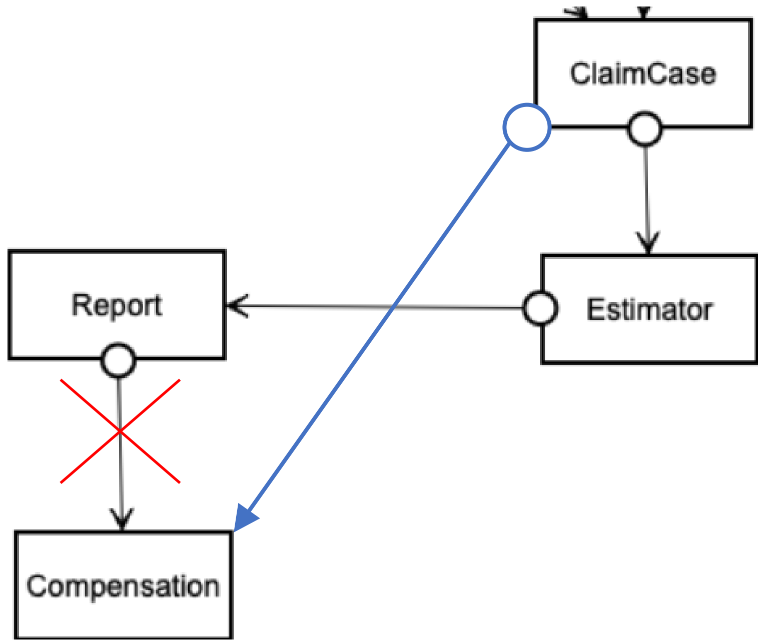


Frequent Mistakes

8 Do not confuse "Use" of information with ED.

" According to the report issued by the estimator, the sum of compensation is calculated."

It's not because the information from the report is used to calculate the compensation, that the object type "Compensation" is ED of the Report.

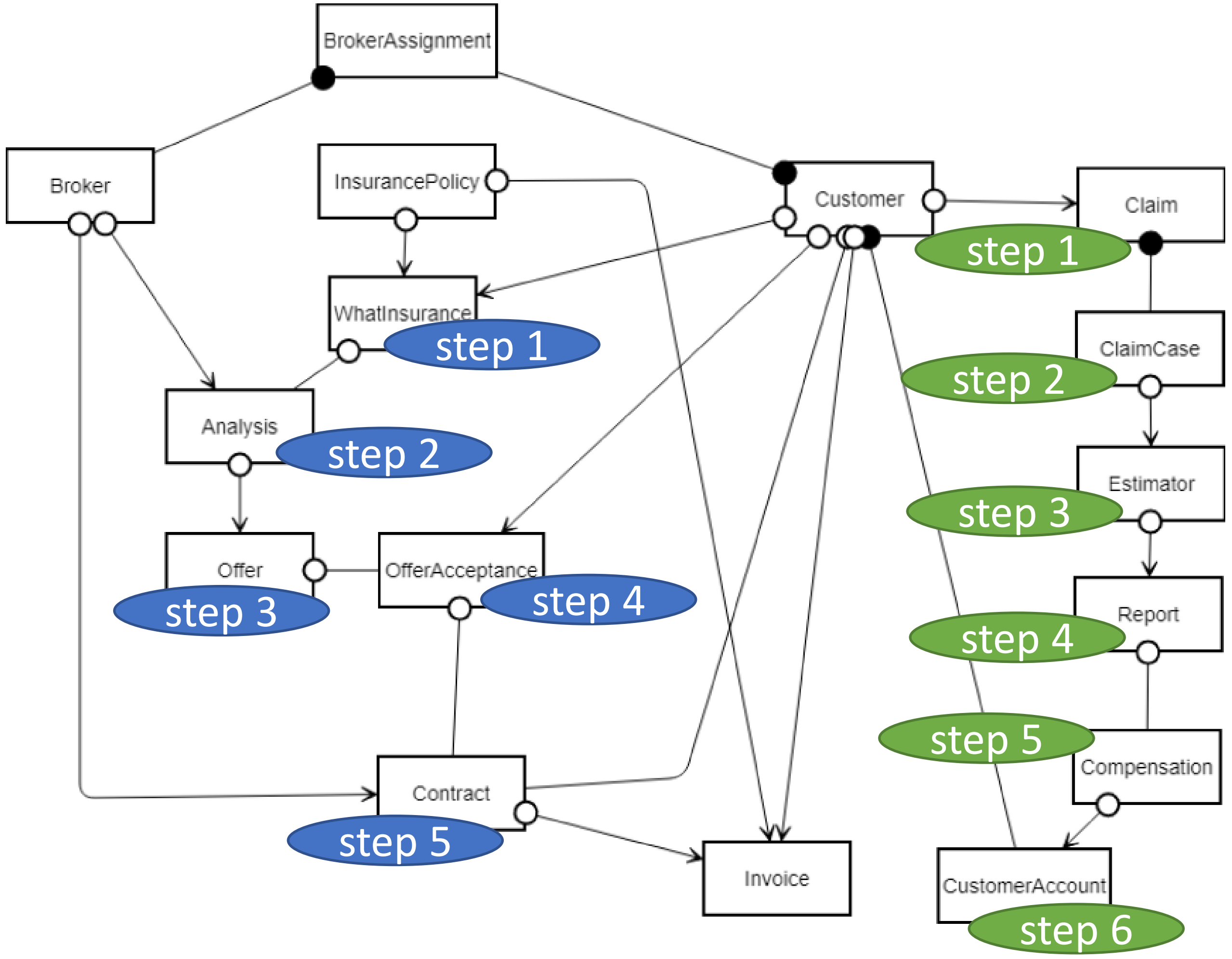


2021 - 2022

"Killer" mistakes

8 Making each step and/or actor in a business process an object type in the EDG resulting in a "chain" of ED object types

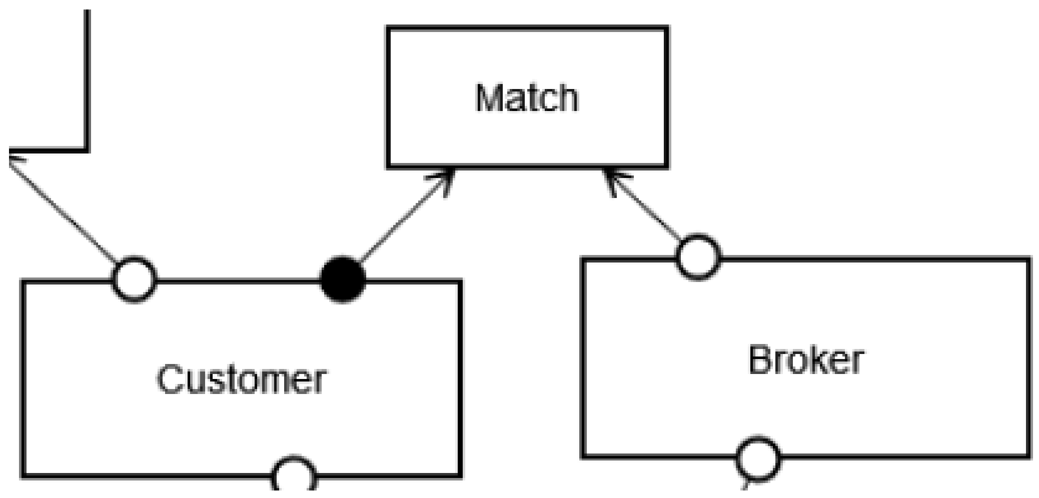
- BP1 = Selling Insurances
- BP2 = Handling Claims



Frequent Mistakes

9 Bad naming

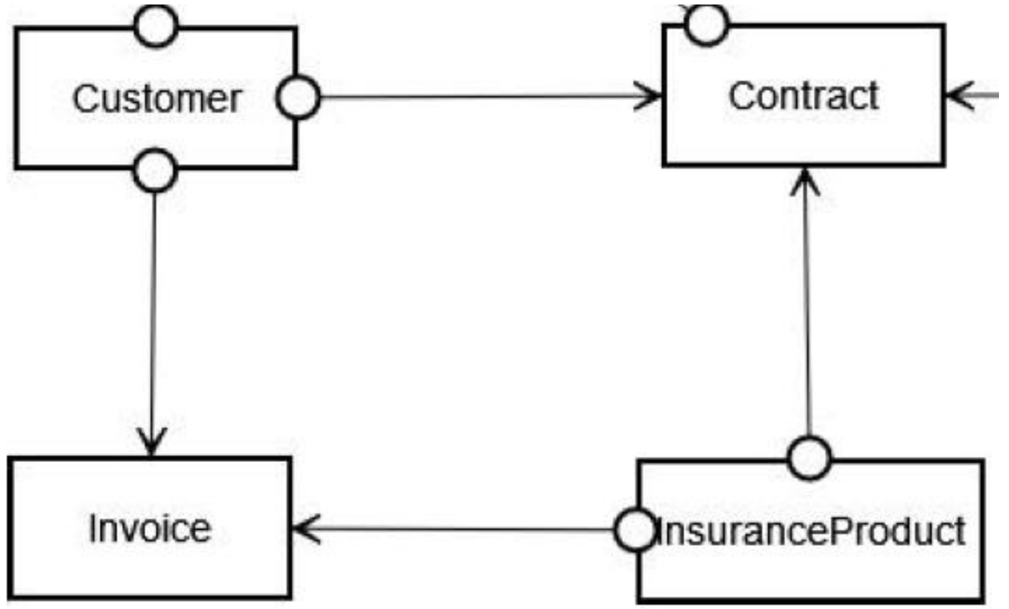
- *E.g.: “Match” instead of BrokerCustomerAssignment*



Frequent Mistakes

10 Not using the “connect to the lowest” pattern

- *E.g.: Invoice is ED on Customer and on InsuranceProduct instead of ED on Contract*



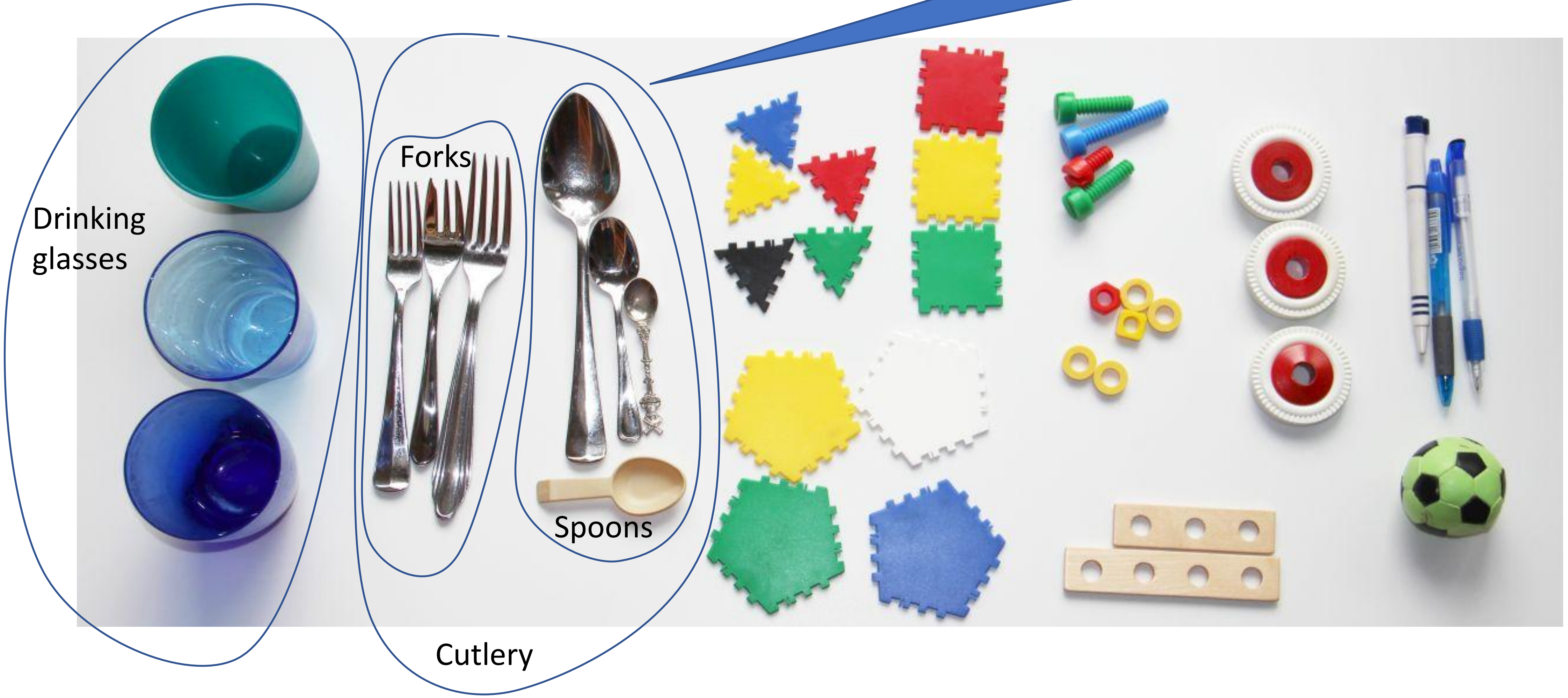
Defining an Object Type

- An object type defines a Class as a set of similar objects
  - “set” in a mathematical sense
  - $A = \{ x \mid x \quad \underbrace{\ldots}_{} \quad \}$

definition of when x belongs to set A
defines the “similarity”, the “kind”

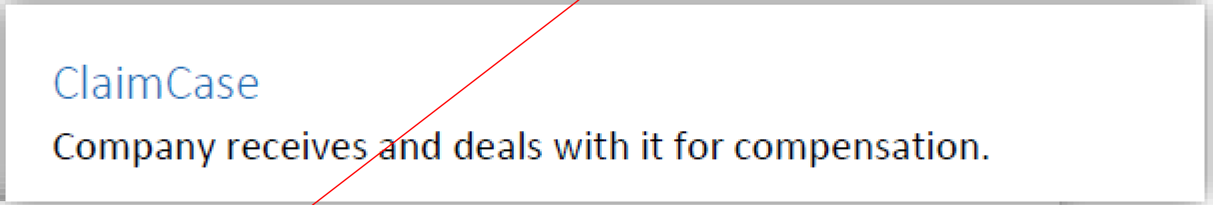
Defining Object Types

The definition of an Object Type should define the characteristics based on which we decide an object to belong to a specific class.



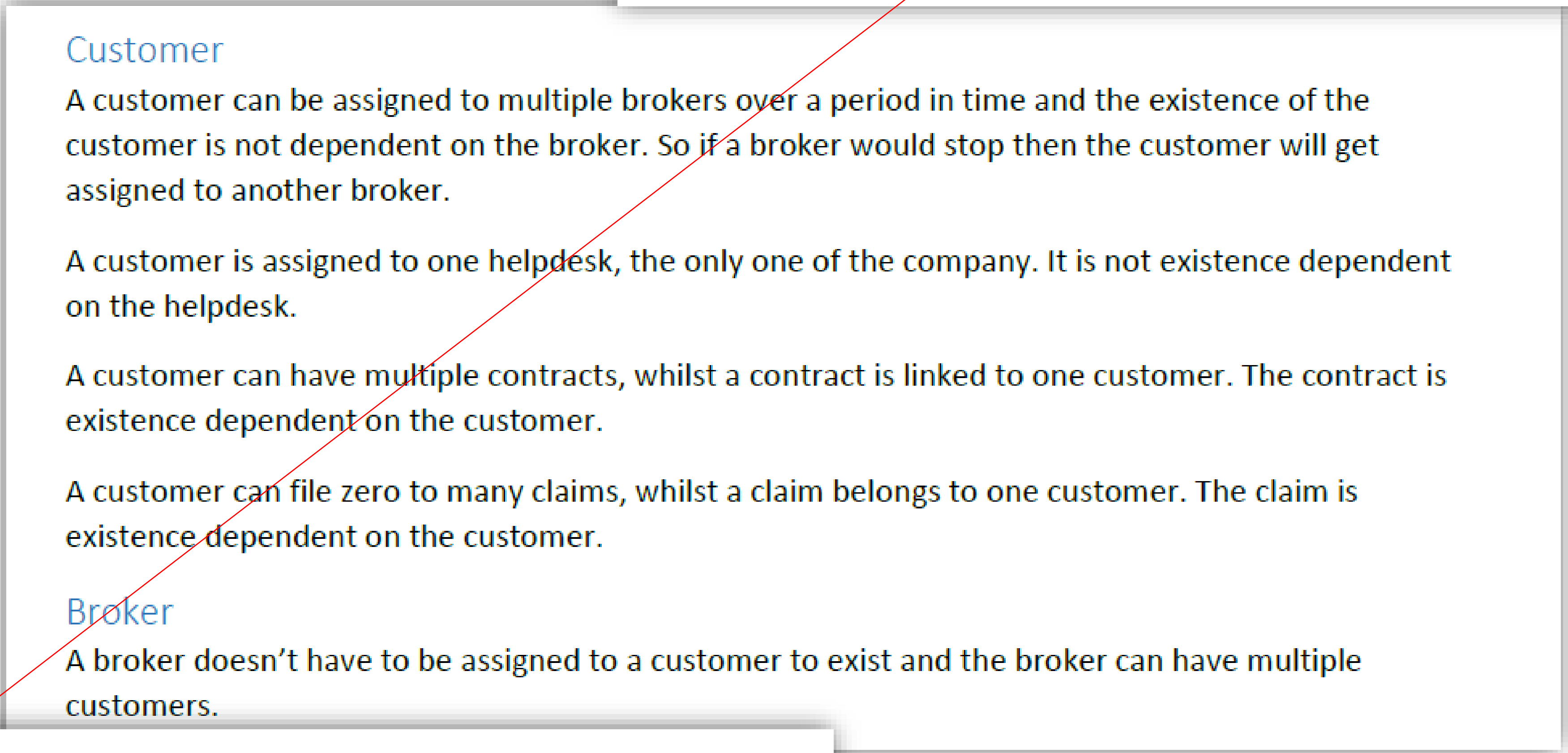
Defining Object Types

NOT GOOD!



ClaimCase

Company receives and deals with it for compensation.



Customer

A customer can be assigned to multiple brokers over a period in time and the existence of the customer is not dependent on the broker. So if a broker would stop then the customer will get assigned to another broker.

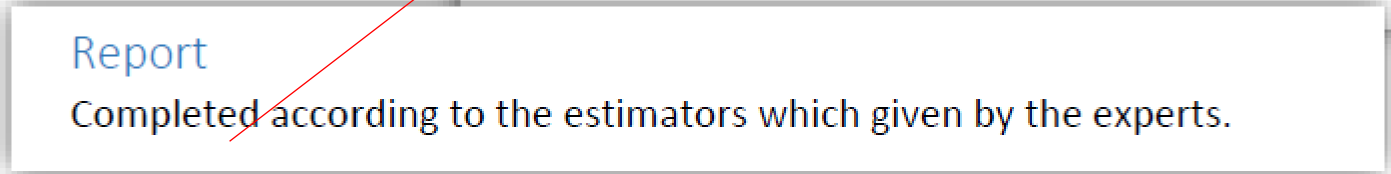
A customer is assigned to one helpdesk, the only one of the company. It is not existence dependent on the helpdesk.

A customer can have multiple contracts, whilst a contract is linked to one customer. The contract is existence dependent on the customer.

A customer can file zero to many claims, whilst a claim belongs to one customer. The claim is existence dependent on the customer.

Broker

A broker doesn't have to be assigned to a customer to exist and the broker can have multiple customers.

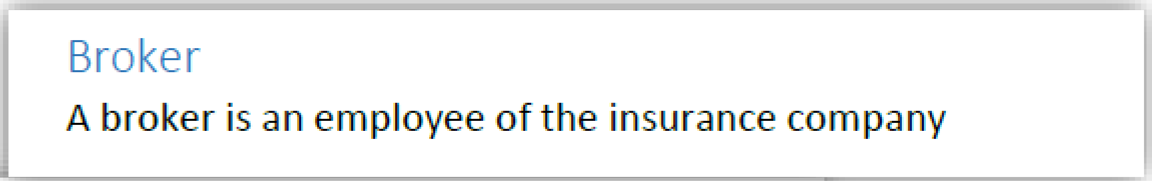


Report

Completed according to the estimators which given by the experts.

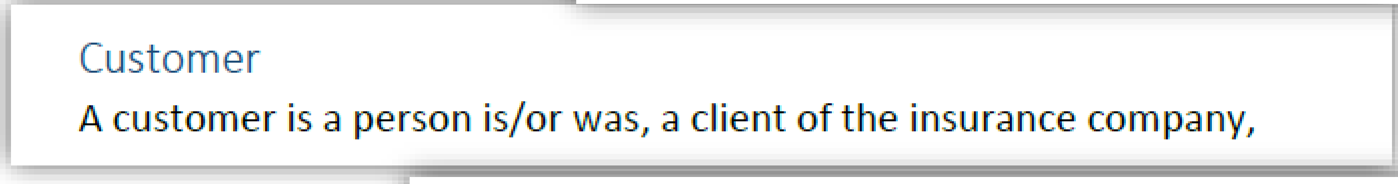
Defining Object Types

- GOOD !



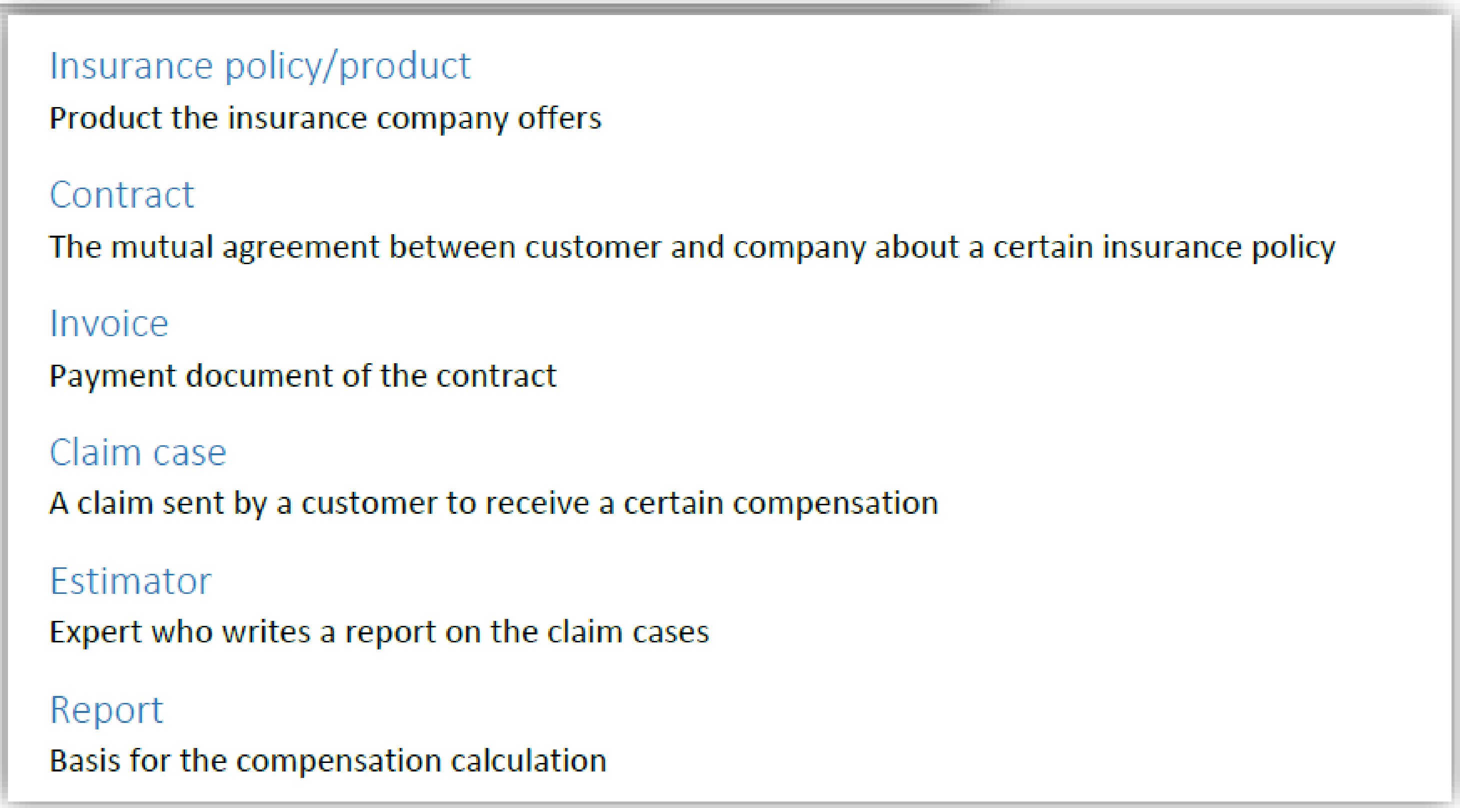
Broker

A broker is an employee of the insurance company



Customer

A customer is a person is/or was, a client of the insurance company,



Insurance policy/product

Product the insurance company offers

Contract

The mutual agreement between customer and company about a certain insurance policy

Invoice

Payment document of the contract

Claim case

A claim sent by a customer to receive a certain compensation

Estimator

Expert who writes a report on the claim cases

Report

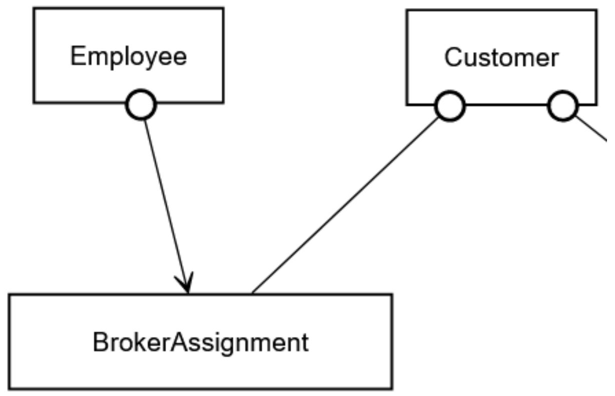
Basis for the compensation calculation

Model Solution Versus Requirements

Object Types

- Broker (Employee)
- Customer
- Broker_Customer_assignment (or Broker if connected to Employee)
- Insurance policy
- Contract
- Offer (optional)
- Invoice
- Claim (optional)
- Claim Case
- Report
- Estimator
- Compensation (Compensation_Decision)
- Compensation_payment (optional)
- Account (optional)

EDG Model solution vs Requirements



"a **broker** is **assigned** to follow the **customer**'s file."

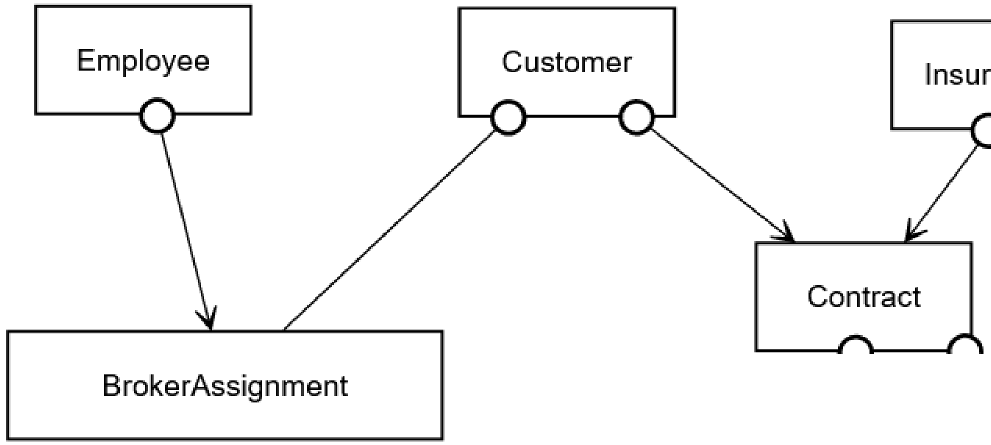
(We consider the Broker to be an employee, and the "Broker" to be a role of this employee

EDG Model solution vs Requirements

Not reflected in EDG (IS layer, BP layer, or addressed through **EL events**):

The **broker** is registered in the system, so that when a customer calls, based on the **contract**, the help desk can immediately trace who is the **customer's** first account manager. The **customer** indicates which type(s) of **insurance policy** they would like to **sign for**, so the broker could, depending on the case, either **assess** the customer's profile on the spot or **send the customer's file for analysis** to the head office. After the **customer's** profile analysis, a preliminary **offer** *(can be modeled as an object type or part of event/state)* on an **insurance product** **is made** to the **customer** either in person or by email.

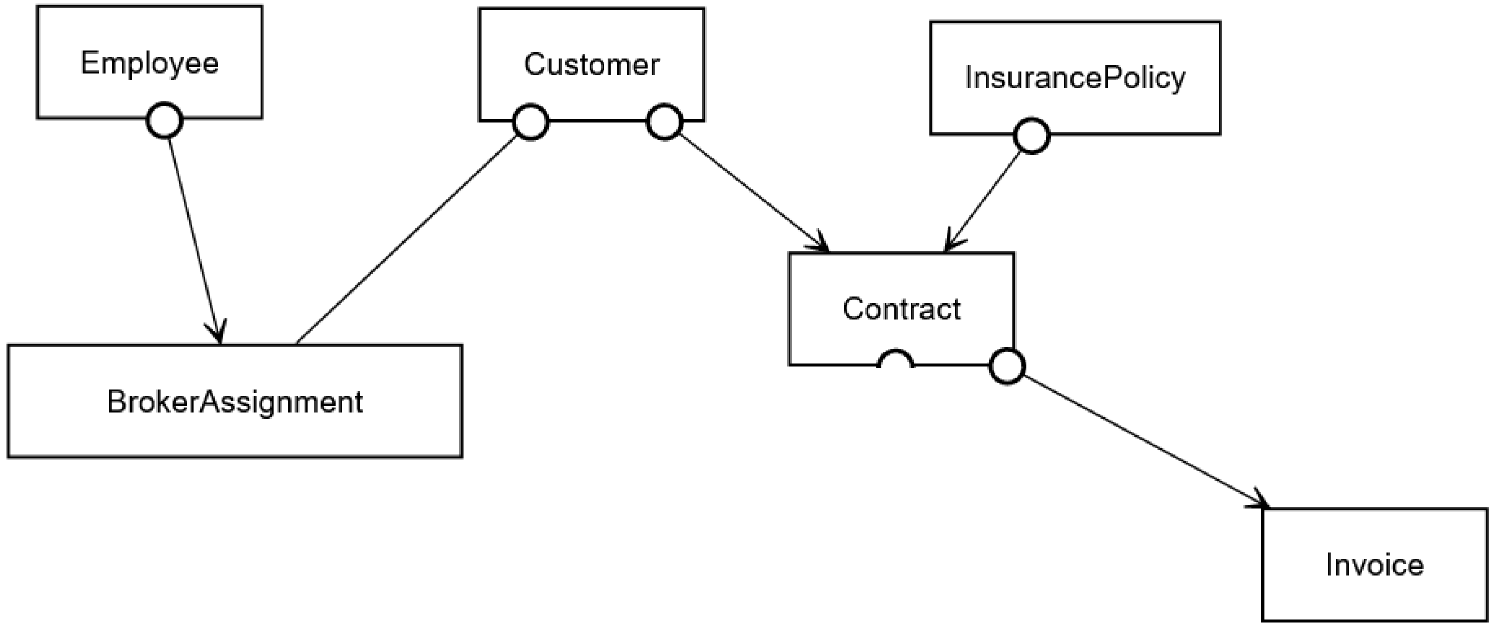
EDG Model solution vs Requirements



"If the **customer** agrees to the offer, a **contract** is generated and signed **by both parties**".

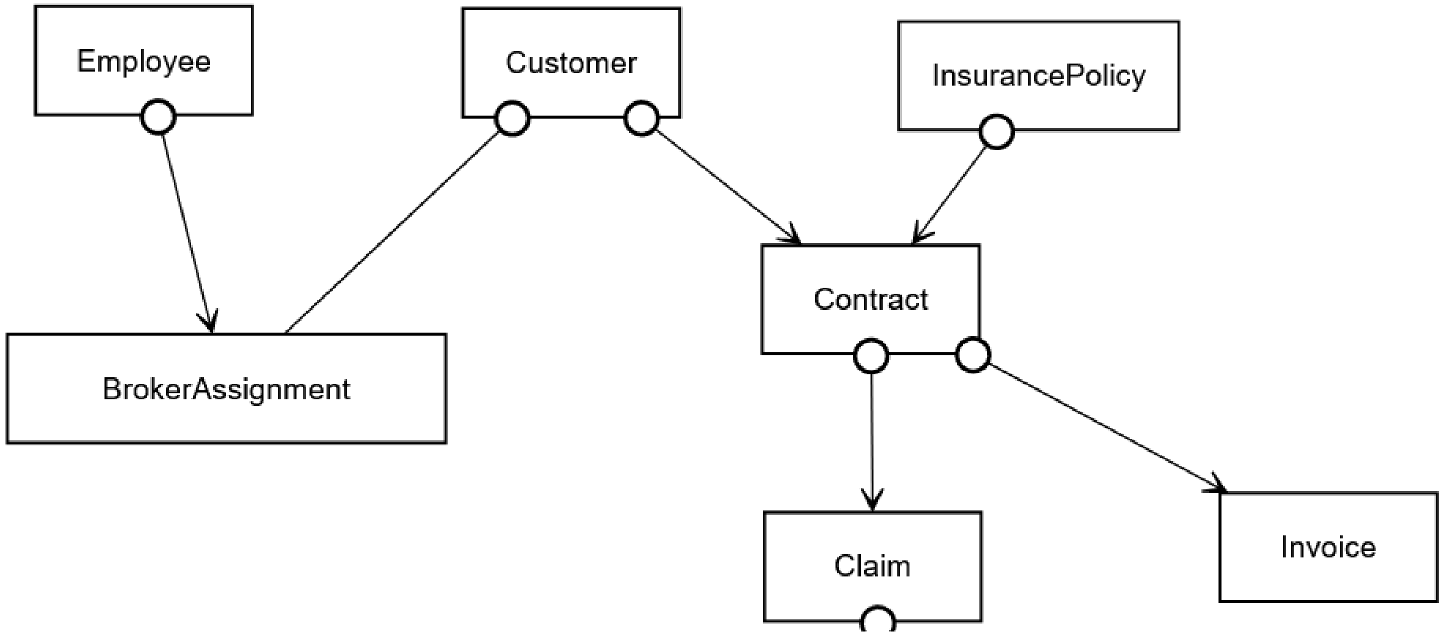
P.S. Modelling a Offer + Contract as one object is also valid, though Contract is sufficient – offer can be a state of Contract.

EDG Model solution vs Requirements



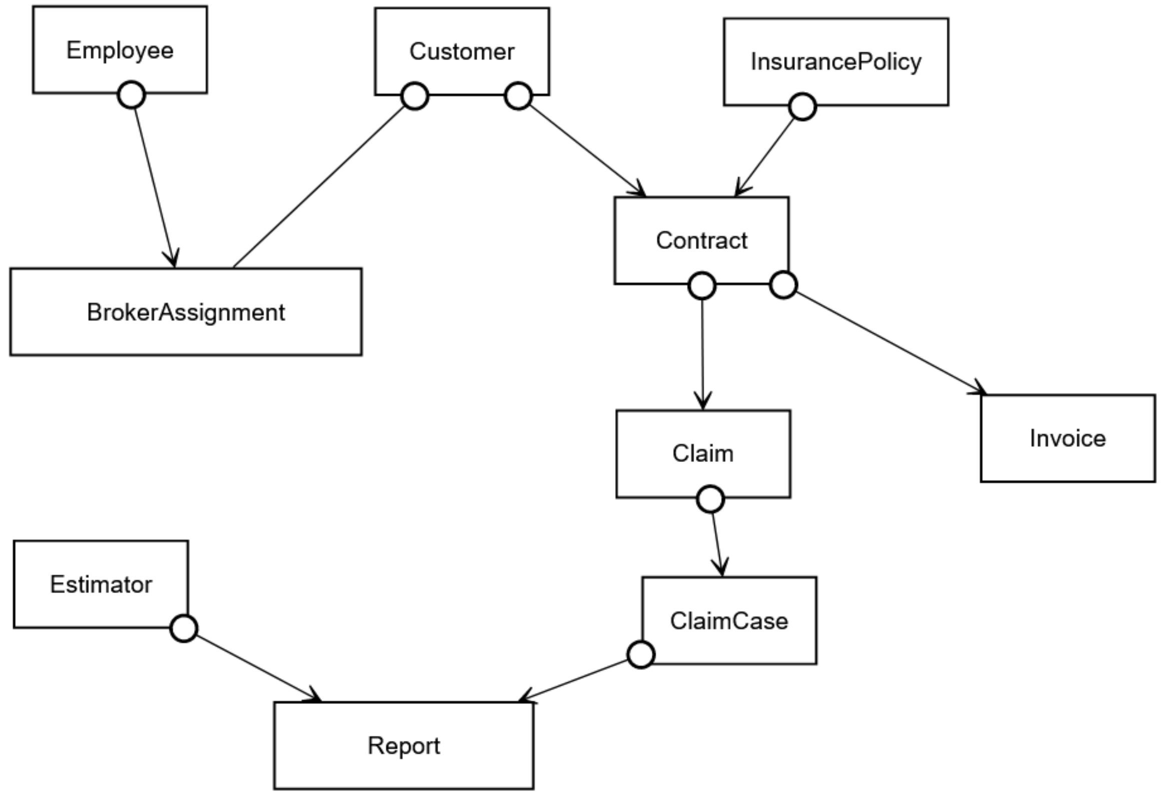
“Afterwards, the **client** is **invoiced** (monthly or yearly – depending on the choice made in the **contract**) according to the price of the **insurance product** they bought.”

EDG Model solution vs Requirements



“In case the insured event happens, a **customer** should send a **claim** for compensation”

EDG Model solution vs Requirements



“Then the company opens a **claim case** and transfers it for assessment by the **estimator**. According to the **report** issued by the estimator, the sum of compensation is calculated.”

P.S. Modelling a Claim + Claim_Case as one object is also valid.
Another alternative correct solution: both Broker and Estimator can be viewed as “Employees” that take different roles.
In this model solution we see Estimators as external experts.

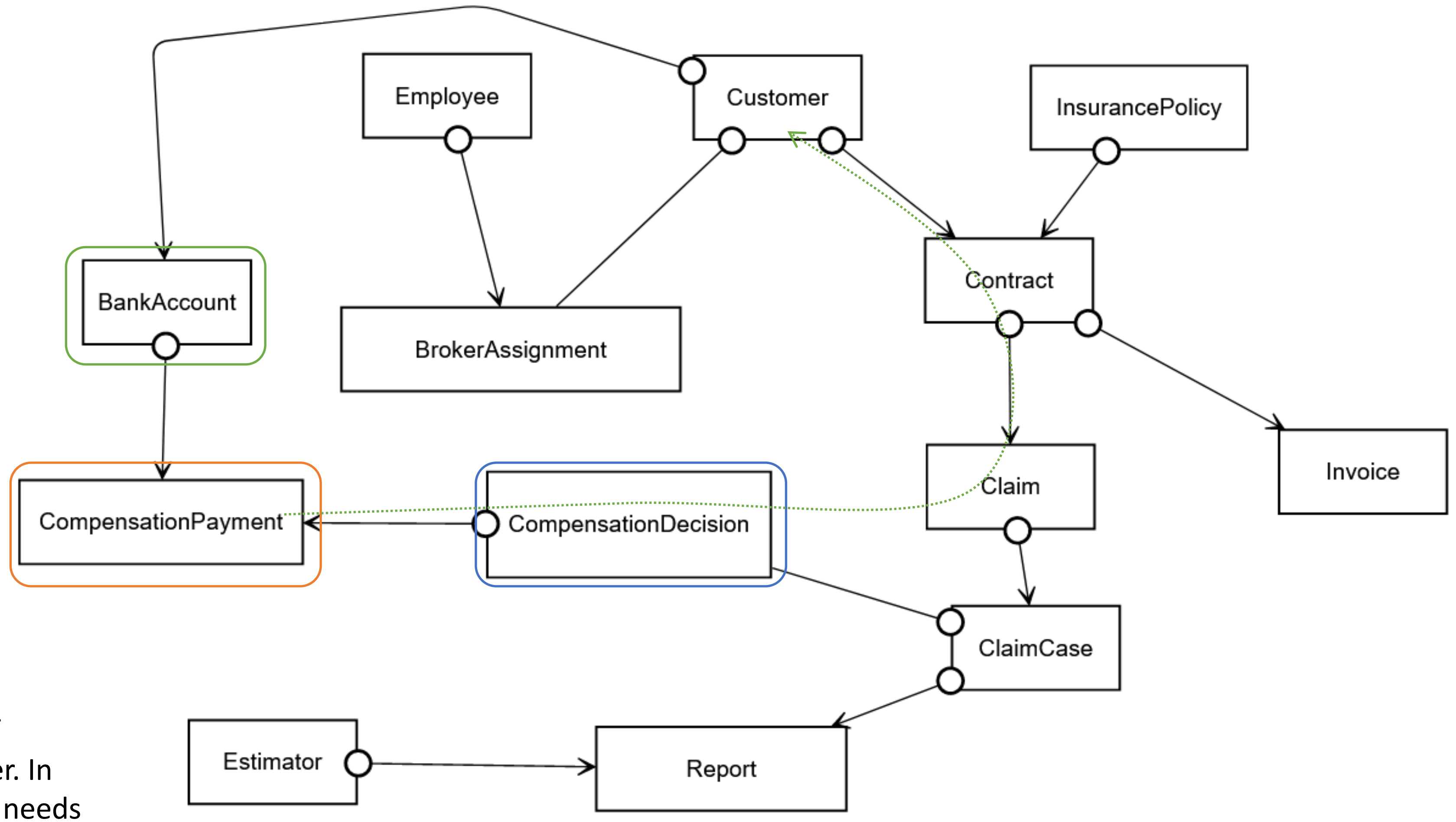
EDG Model solution vs Requirements

“According to the report issued by the estimator, **the sum of compensation is calculated**.
Afterwards, the **compensation** is **paid** to the customer’s **account**.”

The sum of compensations is modelled here as a "**CompensationDecision**". This could be an attribute of the Claim case as well.

The **CompensationPayment** could be an Event, but in practice, for larger cases, several payments are made as proofs are sent by the customer

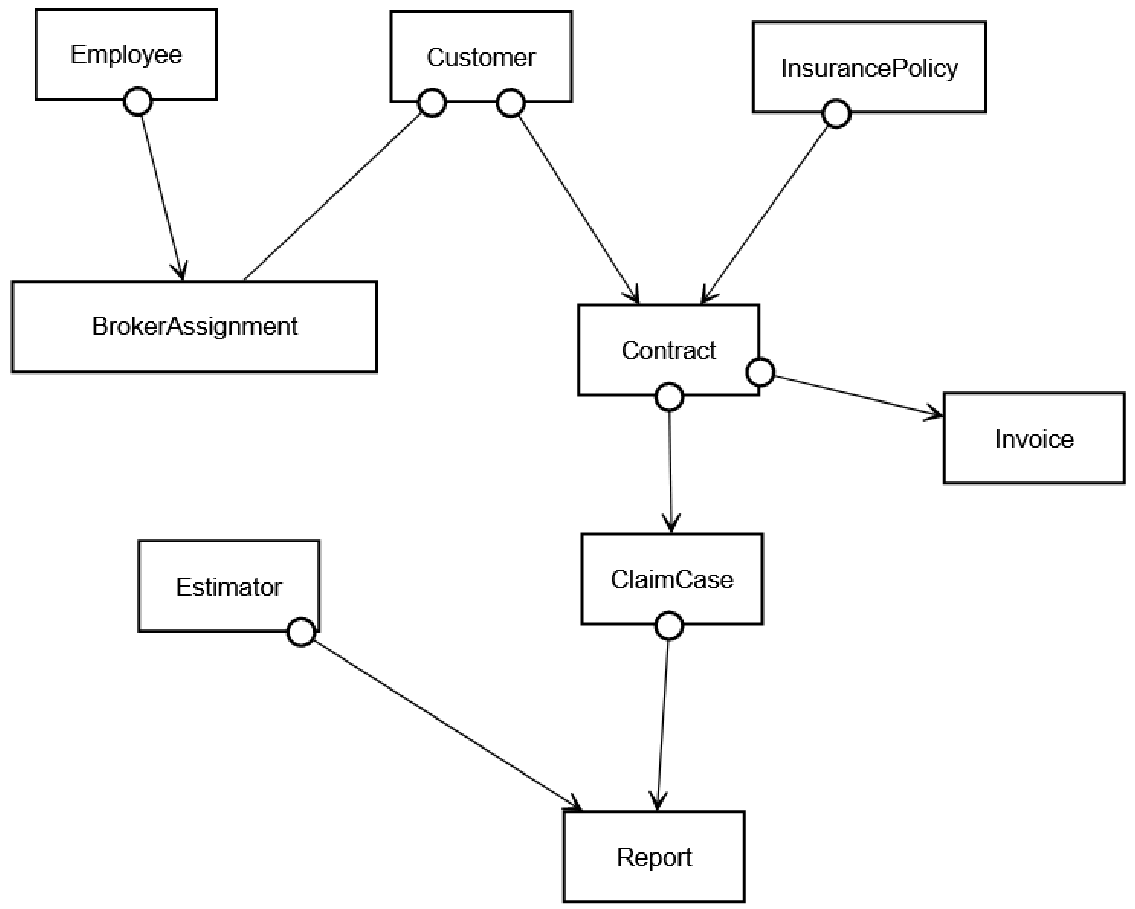
The **BankAccount** could be an attribute of Customer if there is only one per Customer. In that case, no ED relationship to Customer needs to be drawn as the path to Customer can be reconstructed following the green arrow.



Alternative Model Solutions

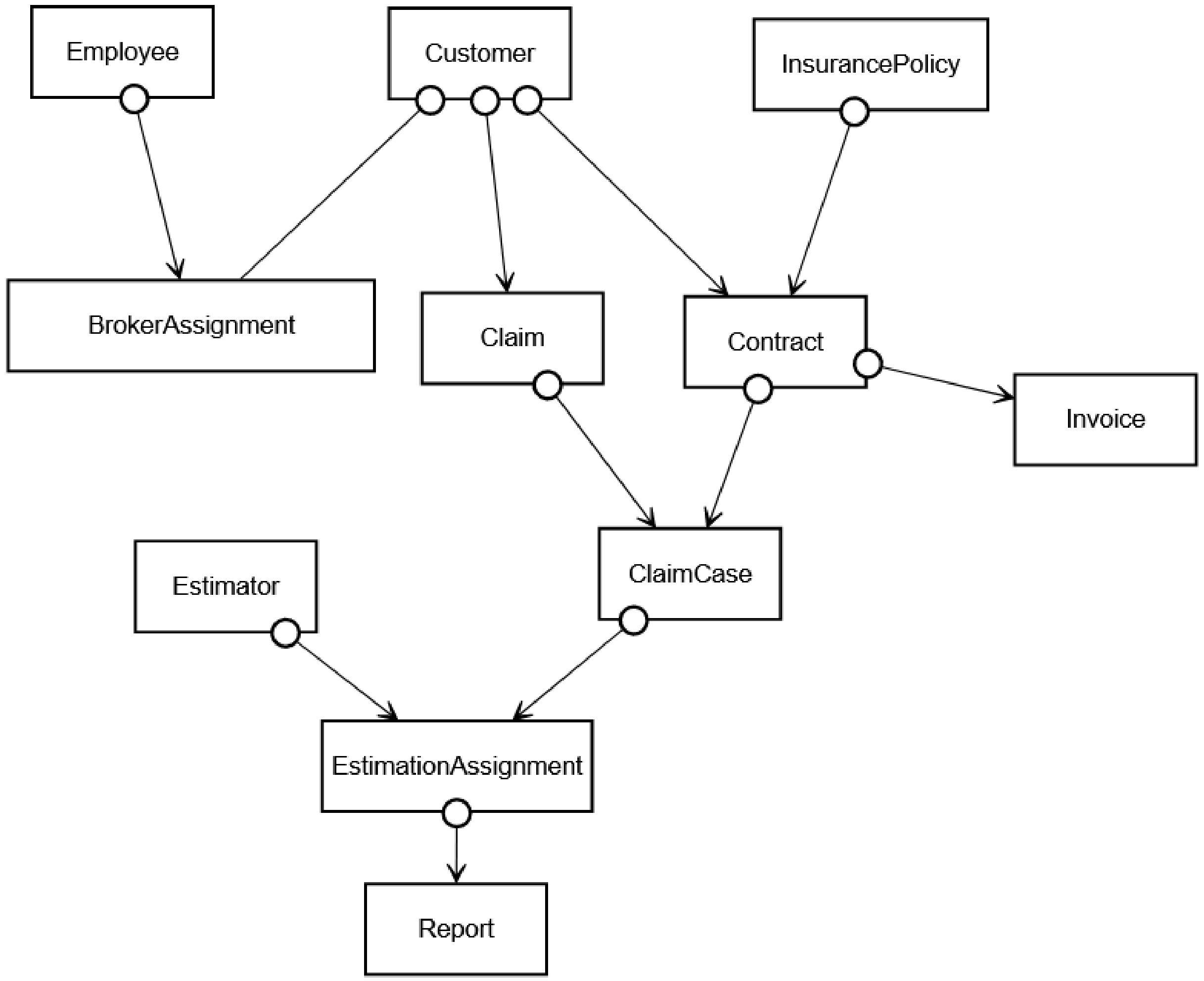
Model Solutions (Basic)

- Alt 1



Model Solutions (Basic)

- Alt 2
  - In this solution, the customer can file a claim that is mapped to one or more contracts afterwards, thus defining Claim Cases.
  - The Estimater has EstimationAssignments for ClaimCases, and can file several reports per each assignment



Approximate grading scheme

- Approximate because some judgment of the overall quality is always made as well.

- Baseline
  - Positive grading: +1 for each correct object type and each correct dependency (Alt1 as a reference scheme --> 17 points to earn)
  - Negative grading: -1 or -2 per error, depending on severity
  - Overall judgment of resulting score + adjustment if needed