Motivation Modelling for Human–Computer Interaction

DR. ELLA ROUBTSOVA

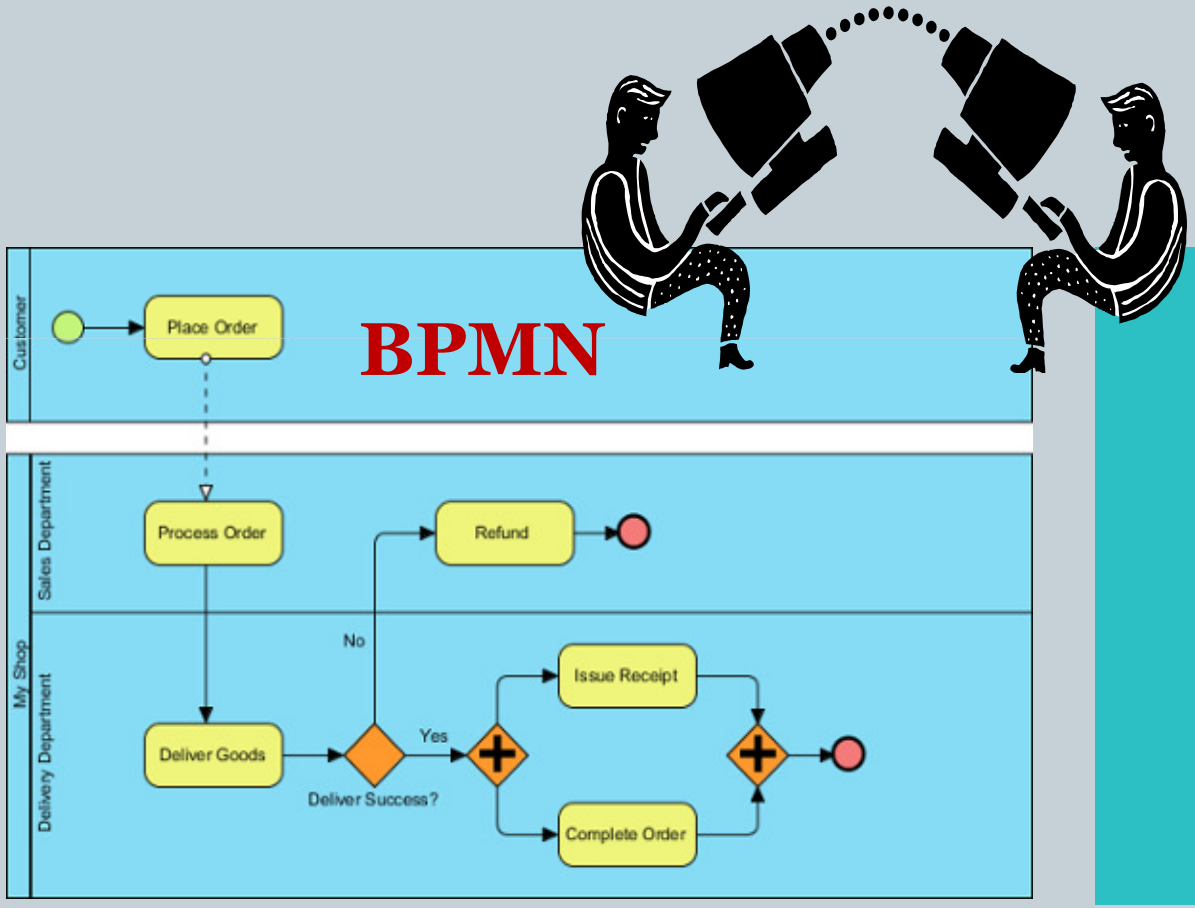
OPEN UNIVERSITY OF THE NETHERLANDS

Business Motivation Model (BMM)

Object Management Group and Business Rules Group

1. **Ends**
2. **Means**
3. **Influences**
4. **Assessments**

Business Process Modelling Notation
Business Motivation Model



BPMN

BMM

Ends

Means

Influences

Assessments

Related Work

- Eric Yu at al. User Requirements Notation (URN) Univ. of Toronto:
  - Goal-Oriented Requirements Language (GRL) based on i* modelling framework (Eric Yu)
  - Use Case Maps (UCM)
- Emmanuel Letier at al. (University College of London) derive event-based transition systems from goal-oriented requirements models:
  - Knowledge Acquisition in autOmated Specification (KAOS)
  - Operations
- Hung Tran Van at al. *Université catholique de Louvain, Belgium* . Goal-Oriented Requirements Animation
  - Goal-oriented notation
  - Goal State Machines (GSMs)

Problems of goal-oriented approaches

"KAOS uses synchronous temporal logics that are interpreted over sequences of states observed at a fixed time rate.

The operational models use asynchronous temporal logics that are interpreted over sequences of states observed after each occurrence of an event.

In order to be semantically equivalent to the synchronous KAOS models, the derived event-based models need to refer explicitly to timing events or include elements of synchronization."

Letier at al

Process + Motivation

P=(S,A,T)

- S {s1, …, sn}
- A {a, b, ….}
- T: t=(si, a, sj)

P = (S; A; C; U)

- C: (a,s) ∈ C - action a is a possible action for P when in state s. Can model.
- U: total mapping C→ S, U(a; si) = sj means if P engages in action a when in state si it will then adopt state sj. U is called Update-model

Want Model

$$P = (S; A; C; U; W)$$

$W \subseteq (A \times S)$, **(a; s)** $\in$ **W** means that action **a** is a wanted action for **P** when in state **s** .

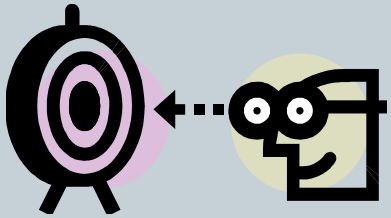
We call relation W the want-model to show its semantic difference from the relation C.

Can and Want Models are independent:

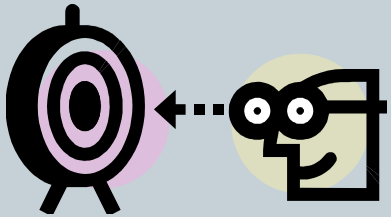
{ can happen; cannot happen } × { wanted; not wanted }

Motivation and Goals

- P = (S; A; C; U; WG1,..., WGn)
- The goals can be OR-composed or AND-composed. What happens with motivation models?



Goal->Objective=Motivation model



Motivation Modelling in Protocol Modelling

- Ends – states and properties on states
- Means – actions
- Influences– protocol machines

Protocol Modelling for Motivation Modelling

- A PM can Read but not modify the state of other PM
- A PM has an associated state function used to derived states.

A DERIVED STATE is a state that is calculated from the states of other machines using the state function associated with the PM

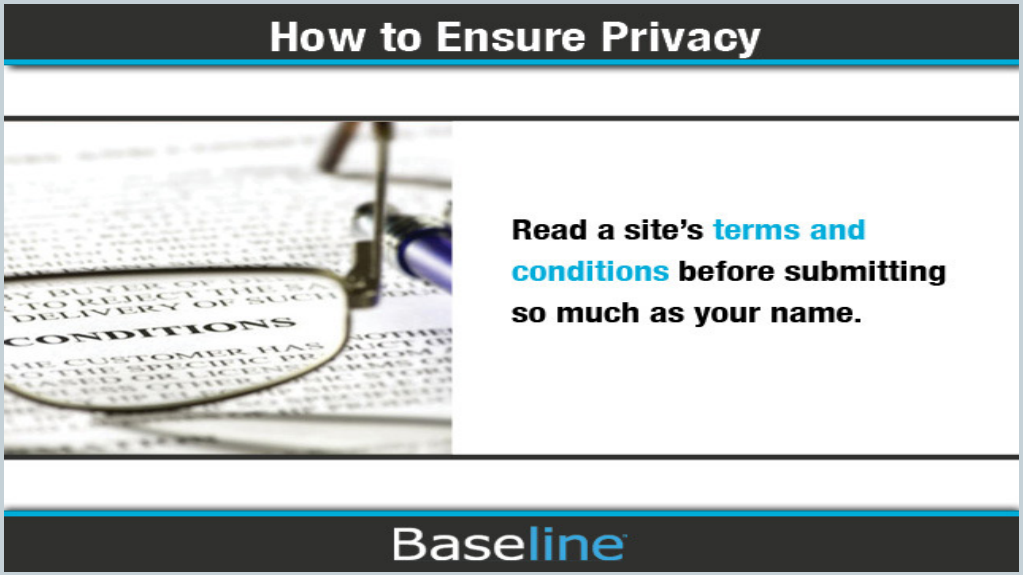
- Different types of PMs have different use of CSP parallel composition

ESSENTIAL (default type) PMs are composed (synchronized) using the CSP parallel composition technique and these machines are used to present the can-update-model, the business process.

DESIRED PMs are not composed using the CSP parallel composition technique.

Insert Credit Curd Number Service

Insert Card Number



Events

- EVENT Instantiate
  ATTRIBUTES Input: Input, Session :String
- EVENT Insert
  ATTRIBUTES Input: Input, Credit Card Number: Integer,
  Decision: Decision,
- EVENT Read
  ATTRIBUTES Person: String, Input: Input, Decision: Decision
- EVENT Cancel
  ATTRIBUTES Input: Input
- EVENT Accept
  ATTRIBUTES Decision: Decision
- EVENT Rethink
  ATTRIBUTES Decision: Decision

Can-Update Model

MODEL InsertCreditCardNumber

OBJECT Input

NAME Session

ATTRIBUTES Session: String, Card Number: Integer

STATES instantiated, cancelled, inserted, read

TRANSITIONS

@new*Instantiate=instantiated,

instantiated*Read=read,

instantiated*Insert=inserted,

instantiated*Cancel=cancelled,

read*Insert=inserted,

read*Cancel=cancelled

OBJECT Decision

NAME Person

ATTRIBUTES Person: String

STATES accepted, not accepted, final

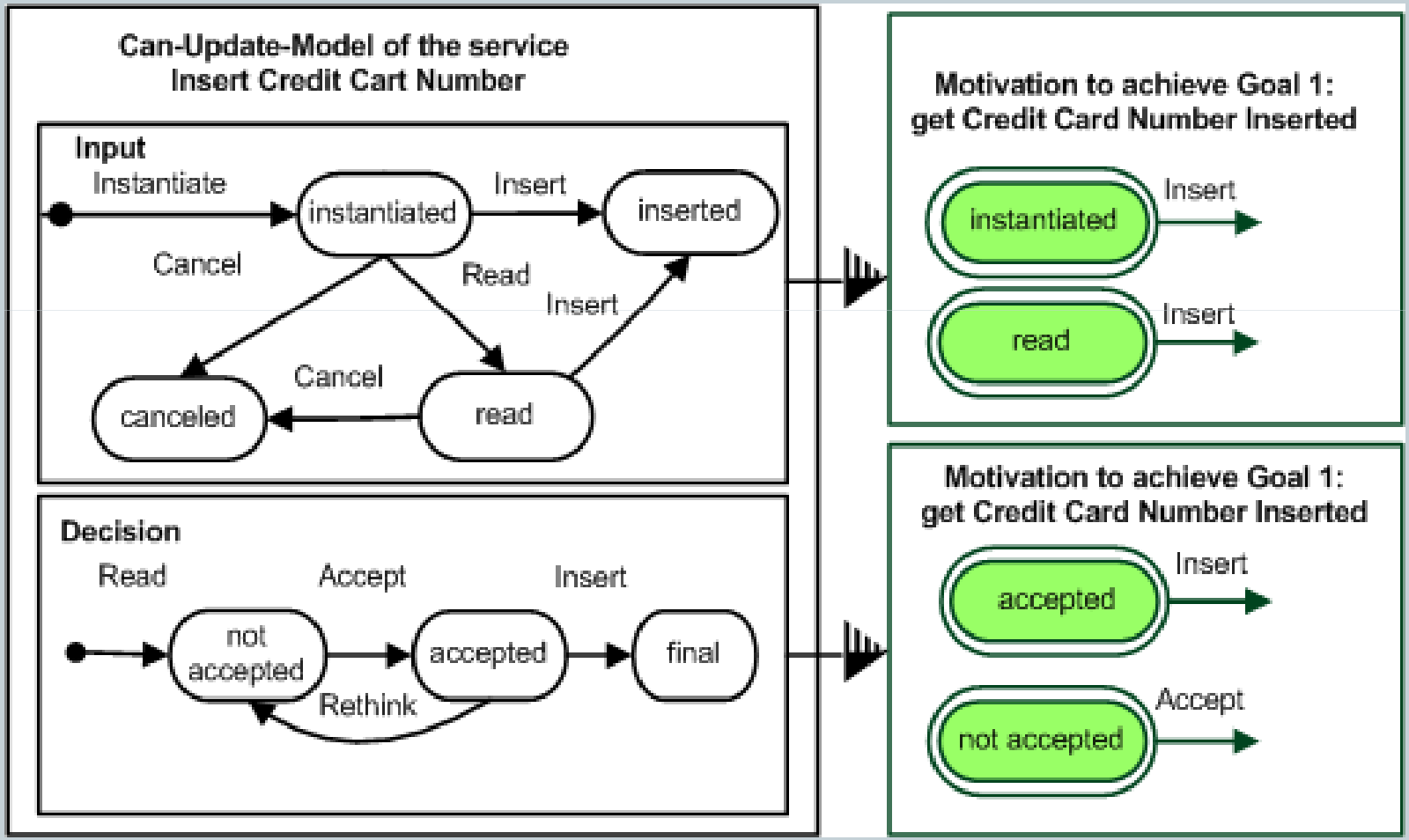
TRANSITIONS @new*Read=not accepted,

not accepted*Accept=accepted,

accepted*Rethink= not accepted,

accepted*Insert=final

Motivation Modelling in Protocol Modelling



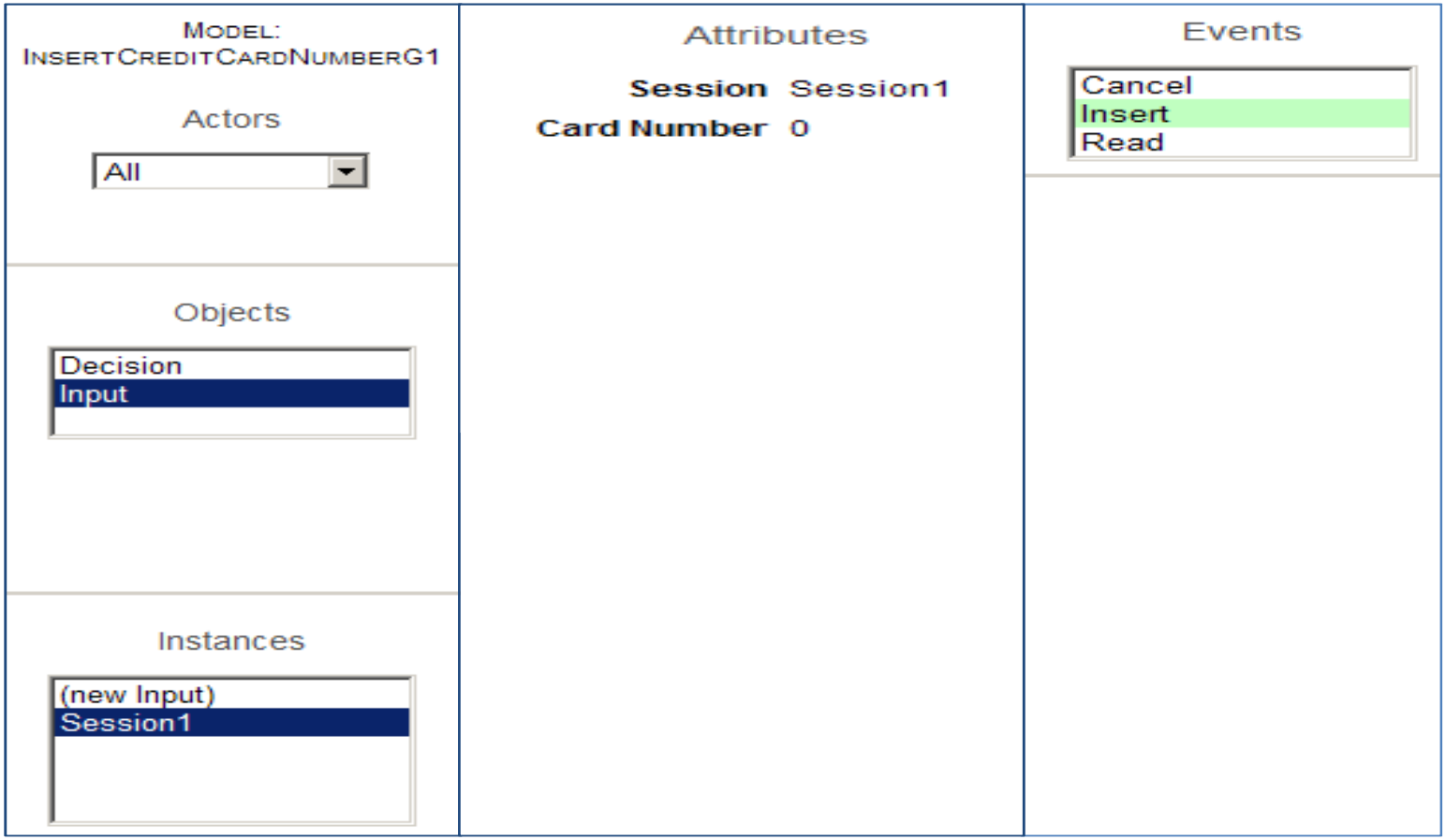
Want Model – Goal Get Card Inserted

BEHAVIOUR !Goal1_getCardInserted_IN_STATE_instantiated_read
TYPE DESIRED
STATES instantiated, read, other
TRANSITIONS instantiated*Insert=@any,
read*Insert=@any
BEHAVIOUR
!Goal1_getCardInserted_IN_STATE_accepted_not_accepted
TYPE DESIRED
STATES accepted, not accepted, other
TRANSITIONS not accepted*Accept=@any,
accepted*Insert=@any

State Function


```
package InsertCreditCardNumberG1;
import com.metamaxim.modelscope.callbacks.*;
public class
Goal1_getCardInserted_IN_STATE_instantiated_read extends Behaviour {
   public String getState () {
      if (this.getState("Input").equals("instantiated"))
        return "instantiated";
      else
      {if (this.getState("Input").equals("read")) return "read";
      else return "other";
      }
}}
```

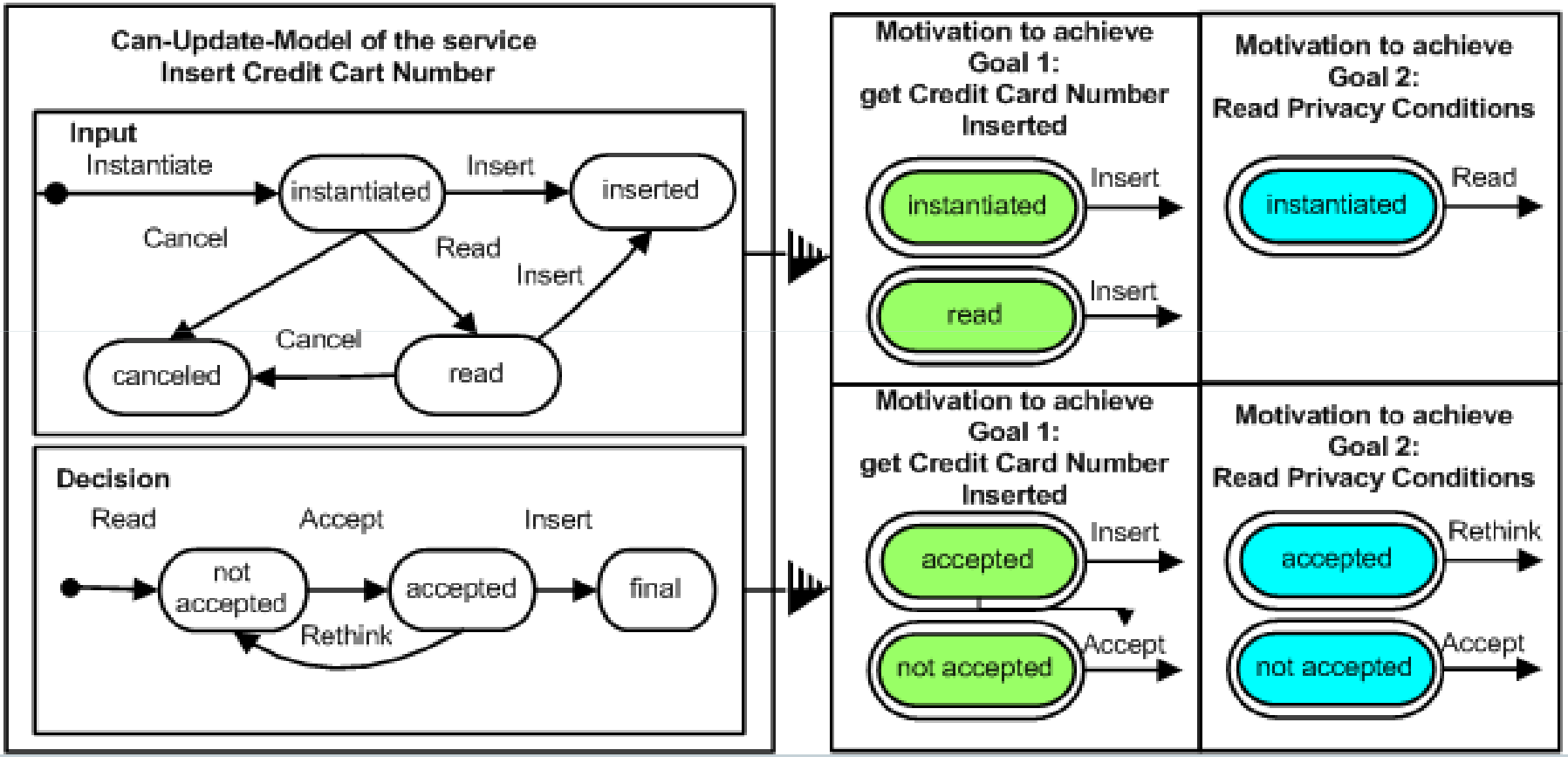
Reflection in the User Interface



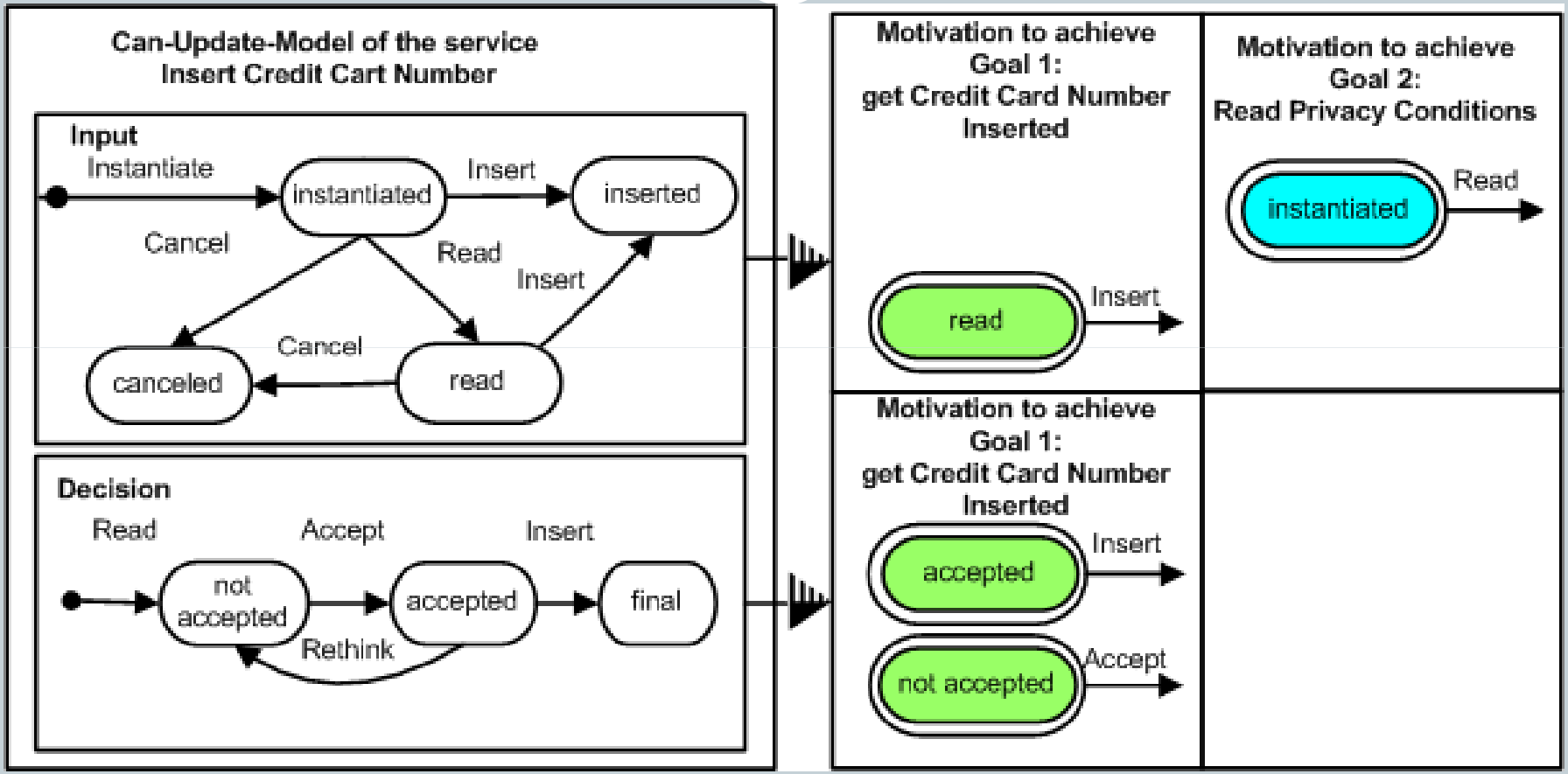
Want Model – Goal Read Privacy Conditions

BEHAVIOUR
!Goal2_ReadPrivacyConditions_IN_STATE_instantiated_read
TYPE DESIRED
STATES instantiated, other
TRANSITIONS instantiated*Read=@any,
BEHAVIOUR
!Goal2_ReadPrivacyConditions_IN_STATE_accepted_not_accepted
TYPE DESIRED
STATES not accepted, accepted, other
TRANSITIONS not accepted*Accept=@any,
accepted*Rethink=@any

OR- composition



AND-composition



Application

- Justification of the user interface design
- Generated of elements of the user interface: MDSD
- Reuse of services with different goals.
- Reflection goals in models
- Identification of contradicted goals and requirements

Future Work

- Development of a practical and evolutional method for motivation modelling in large protocol models with different sets of goals
- The most interesting direction for future work is connecting web services on the basis of matching motivation models.

Creating collaborative businesses from web services with matching motivation models promises the possibility of building more effective e-businesses.