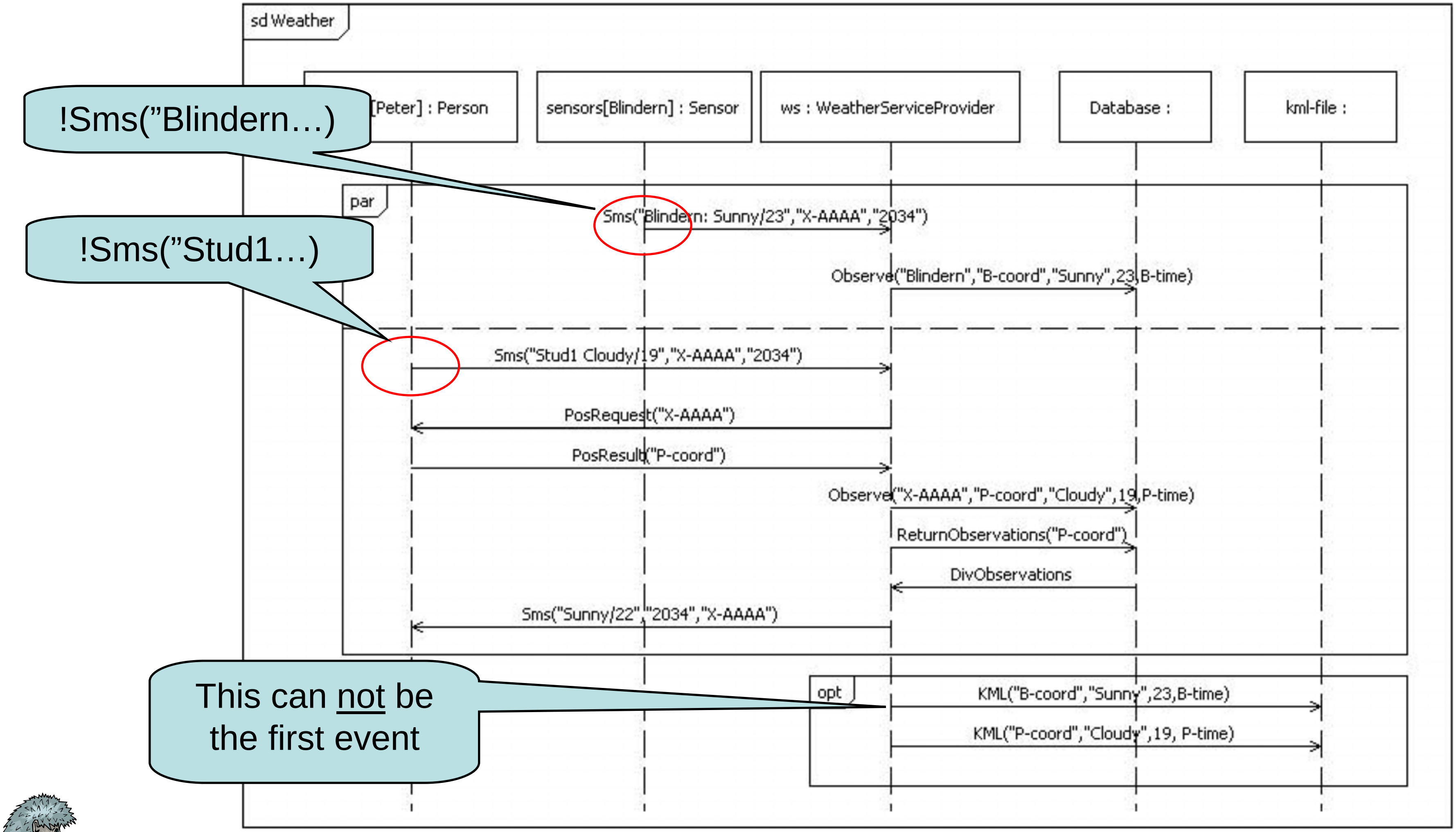
Oblig 1 suggested solution with some comments

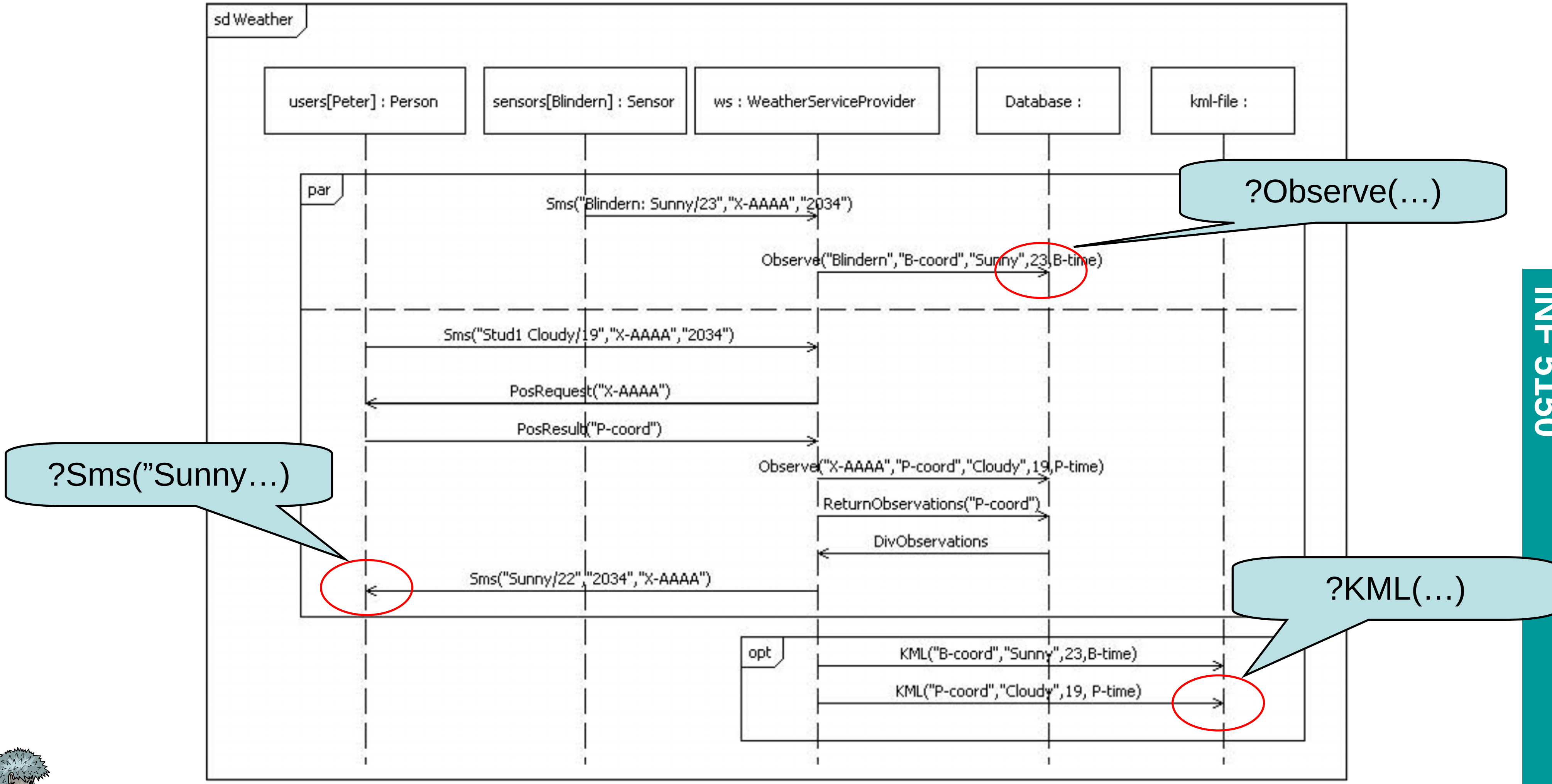
071019

a) I First events

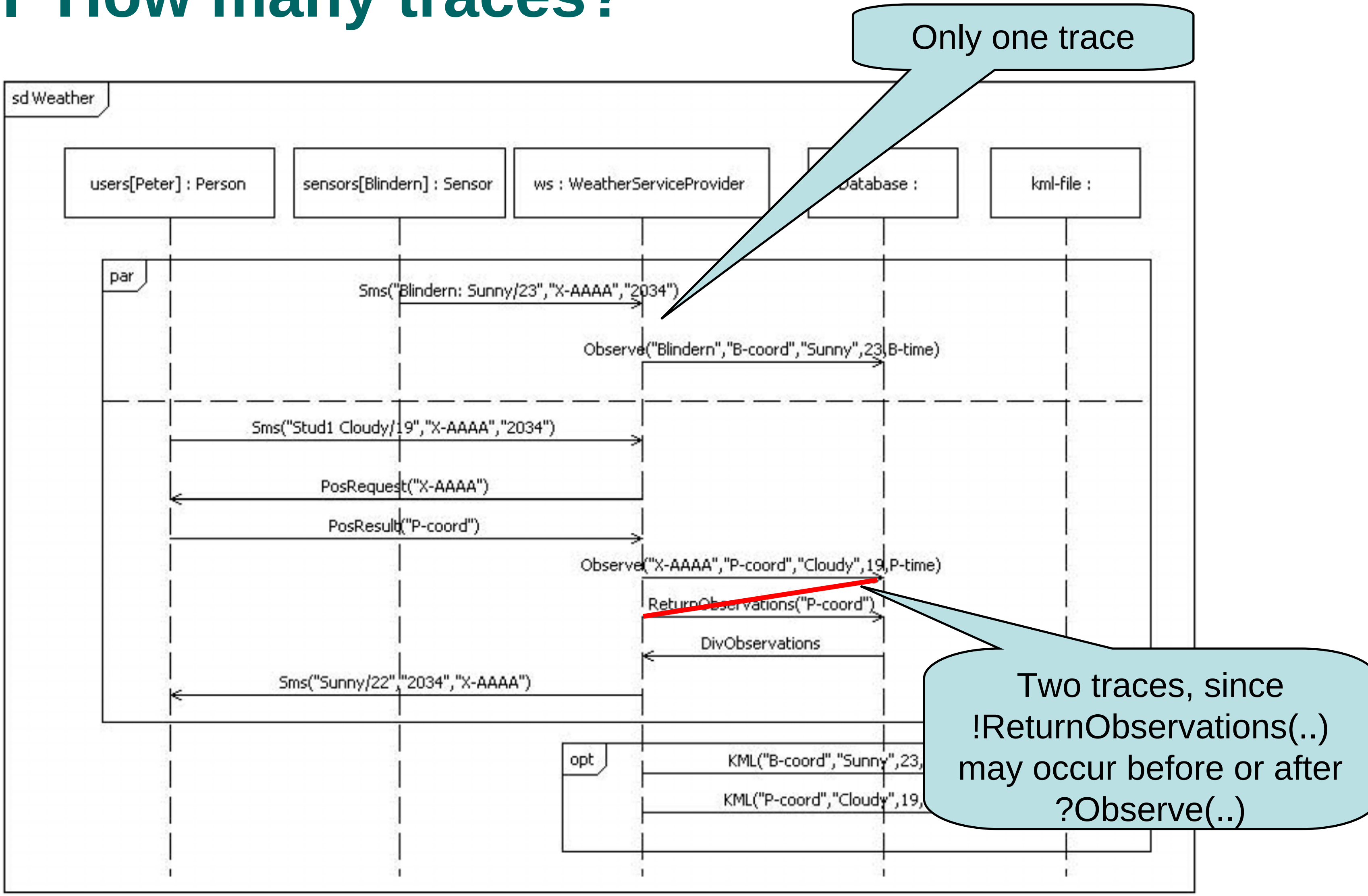
Remember to distinguish between send- and receive-events.
Use ! and ?



a) II Last events

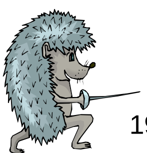


b) I+II How many traces?



b) III+IV

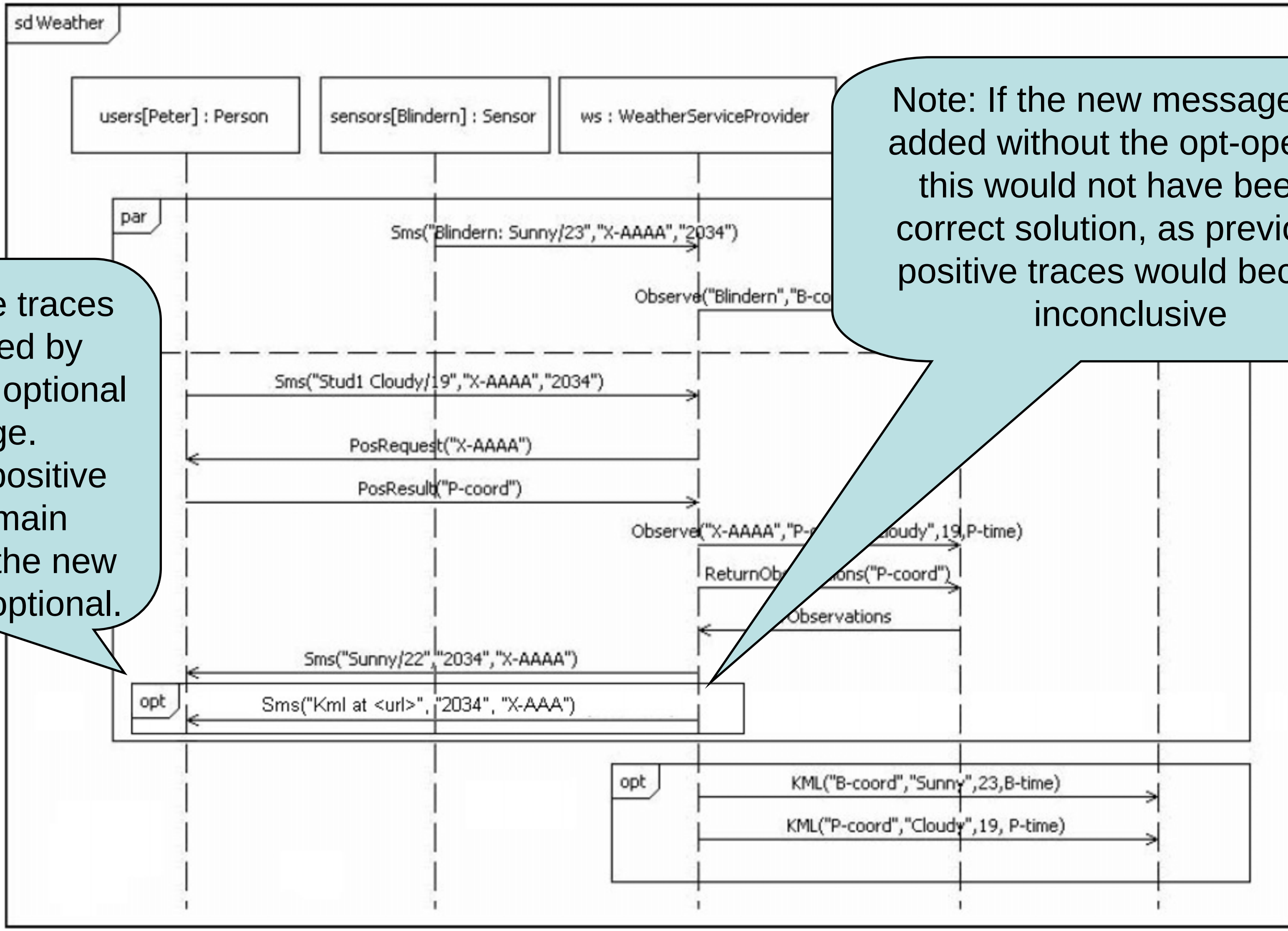
- III: How many negative traces?
  - None, since we have not used any of the operands that define negative behavior: neg/veto, refuse, assert, guard
- IV: How many inconclusive traces?
  - Infinitely many, because
  - Inconclusive = $\mathcal{H} \setminus (p \cup n)$
  - The number of traces in $\mathcal{H}$ is infinite
    - This would in fact hold even if we had only one event *e*, since we would then have ={<>,<*e*>,<*ee*>,<*eee*>,…}
  - p∪n contains only a finite number of traces (n is in fact empty)



c) Refinement

- ”Pure” supplementing: Do supplementing without doing anything else
- ”Pure” narrowing: Do narrowing without doing anything else
- The original specification contains only one interaction obligation (there is no xalt)
- The task is best solved by doing supplementing/narrowing of this single interaction obligation
  - Do not introduce xalt in order to do supplementing!
    - (If you do this, you need to ensure that <u>all</u> the new interaction obligations are pure supplementings of the original)

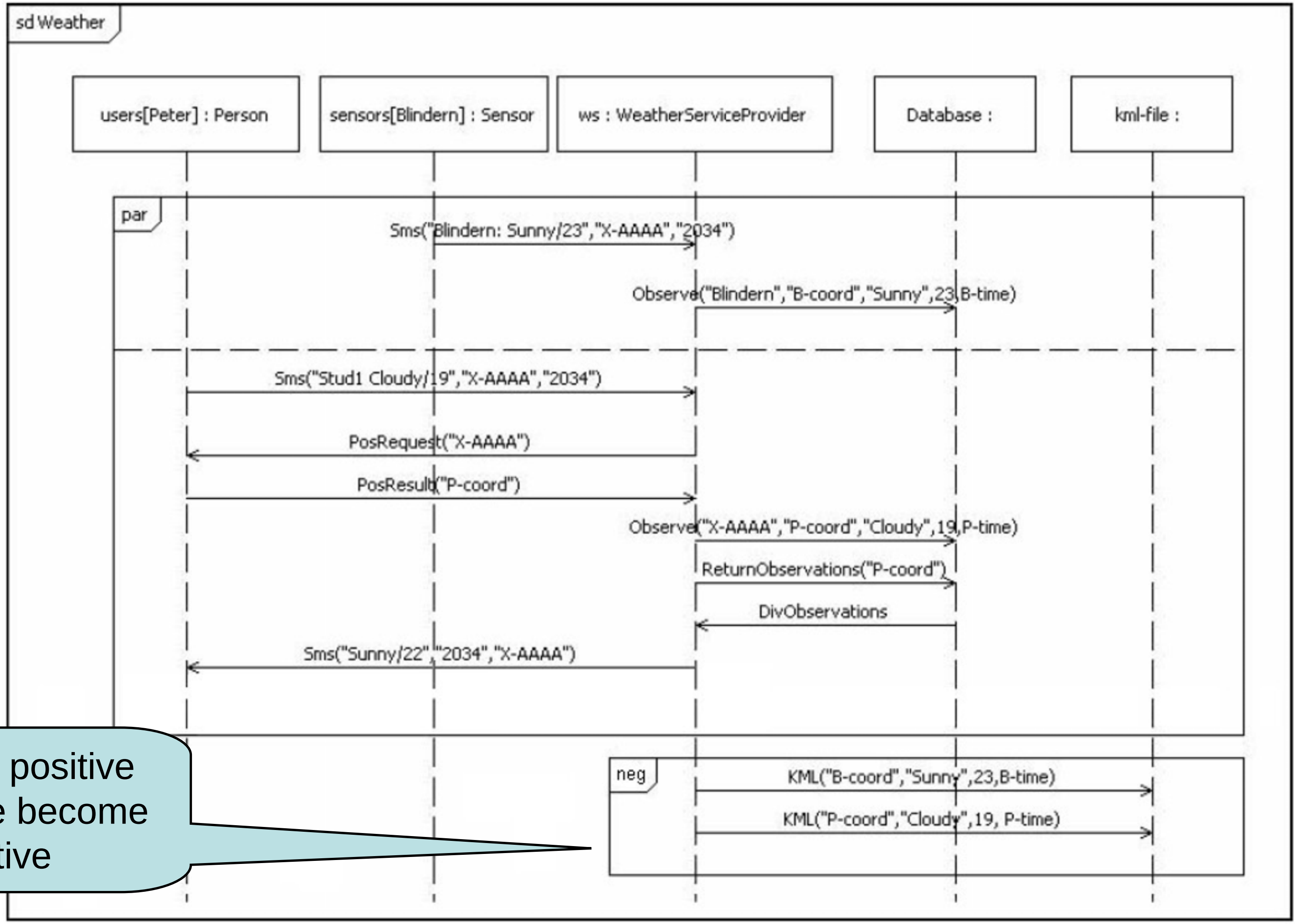
c) I Pure supplementing (the solution of a student)



New positive traces are obtained by including the optional message. Previously positive traces remain positive, as the new message is optional.

Note: If the new message was added without the opt-operator this would not have been a correct solution, as previously positive traces would become inconclusive

c) II Pure narrowing



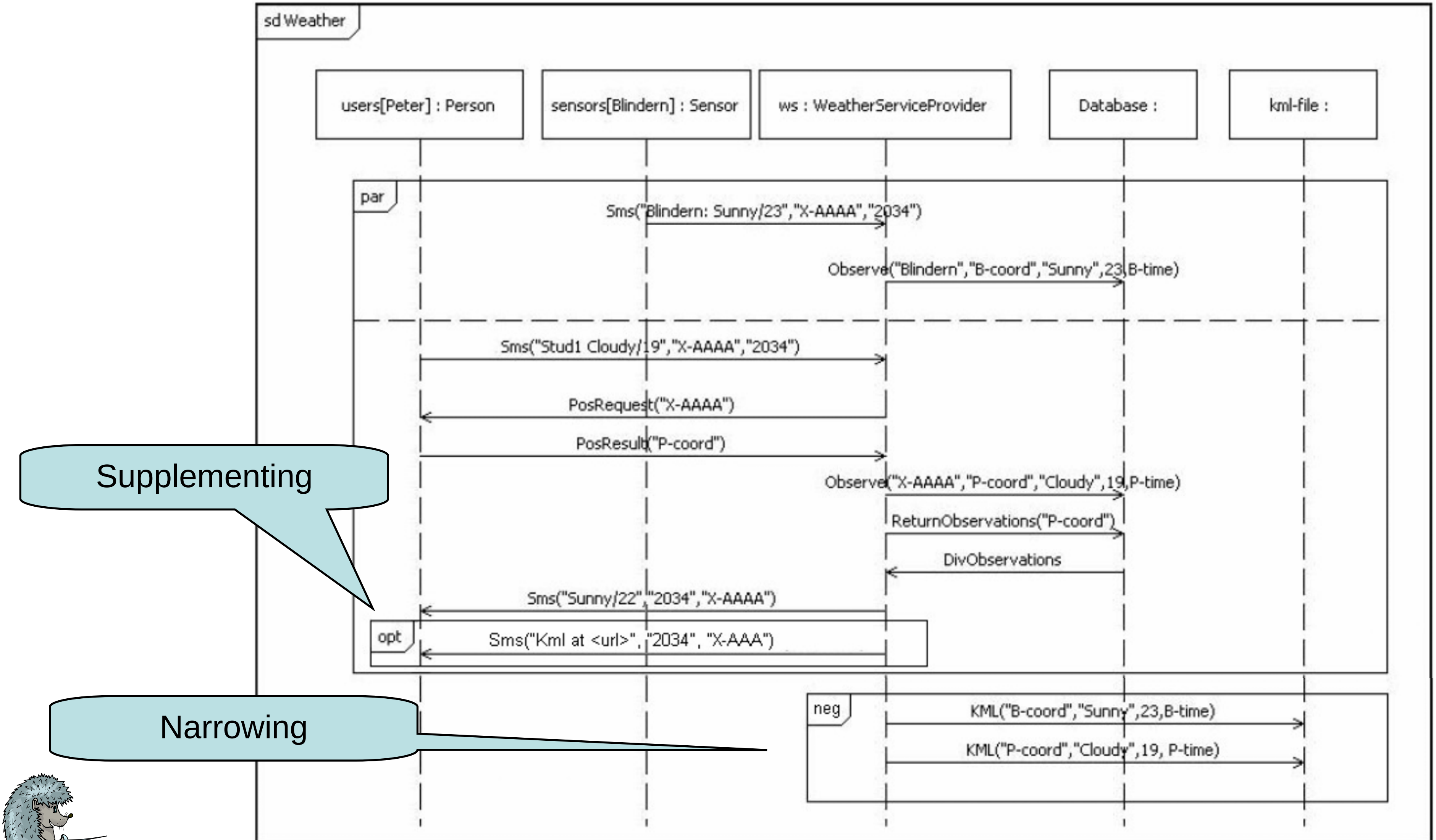
Previously positive traces have become negative

Some comments on opt and neg

- There is no point in enclosing (neg …) directly in opt, since [[opt (neg d)]]=[[neg d]]
  - Example: Let [[d]]={ (p,n) }
  - [[opt (neg d)]] = [[skip alt (neg d)]] = [[skip]] $\uplus$ [[neg d]] = $\{ (\{<>\},\emptyset) \} \uplus \{ (\{<>\},p\cup n) \} = \{ (\{<>\},p\cup n) \}$ = [[neg d]]

- Enclosing (opt …) directly in neg means that the empty trace becomes both positive and negative
  - <> is positive in the interaction obligation(s) of [[opt d]]. It is therefore made negative (as well as positive) by the neg operator
  - [[opt d]] = [[skip alt d]] = $\{ (\{<>\}\cup p,n) \}$
  - [[neg (opt d)]] = $\{ (\{<>\}, \{<>\}\cup p\cup n) \}$

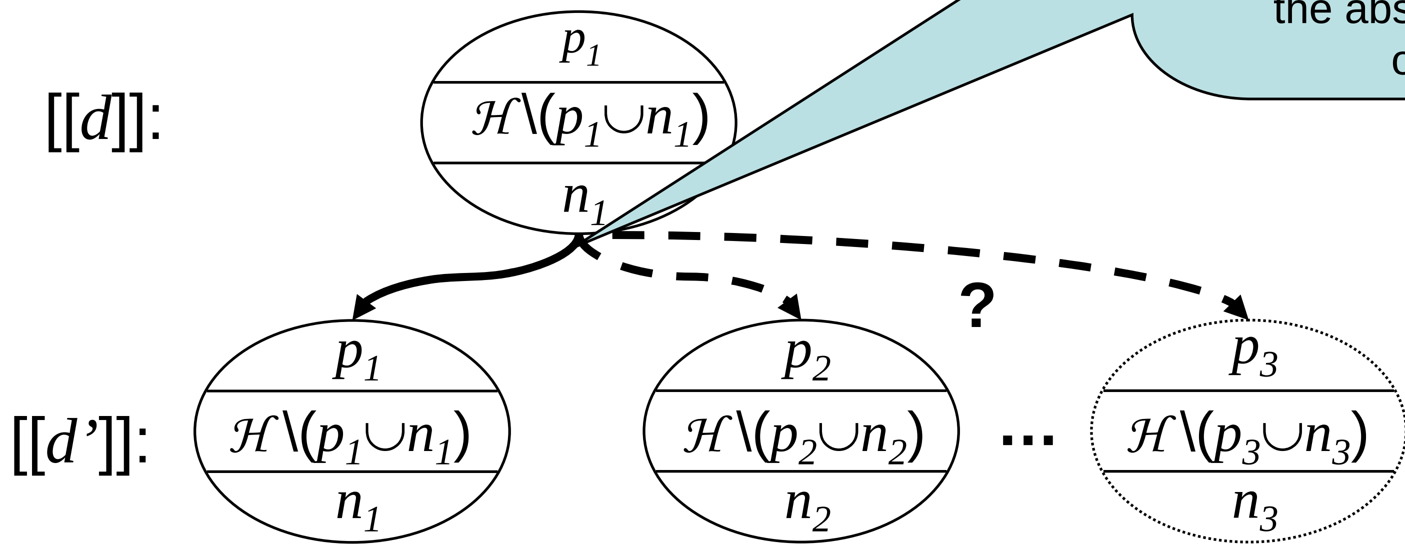
c) III Supplementing and narrowing



d) I General refinement

General refinement holds for certain, as we know that the only interaction obligation in [[*d*]] is refined by at least one interaction obligation in [[*d'*]]

(which in this case is identical to the abstract interaction obligation)



Unknown number and contents of new interaction obligations
(the new xalt operand may contain more xalt operands)

d) II Limited refinement

Limited refinement *may* hold – *if* the new interaction obligation(s) happen to be refinements of the (only) abstract interaction obligation. We do not know whether this is the case.

(Answers stating that limited refinement does not hold have been accepted.)