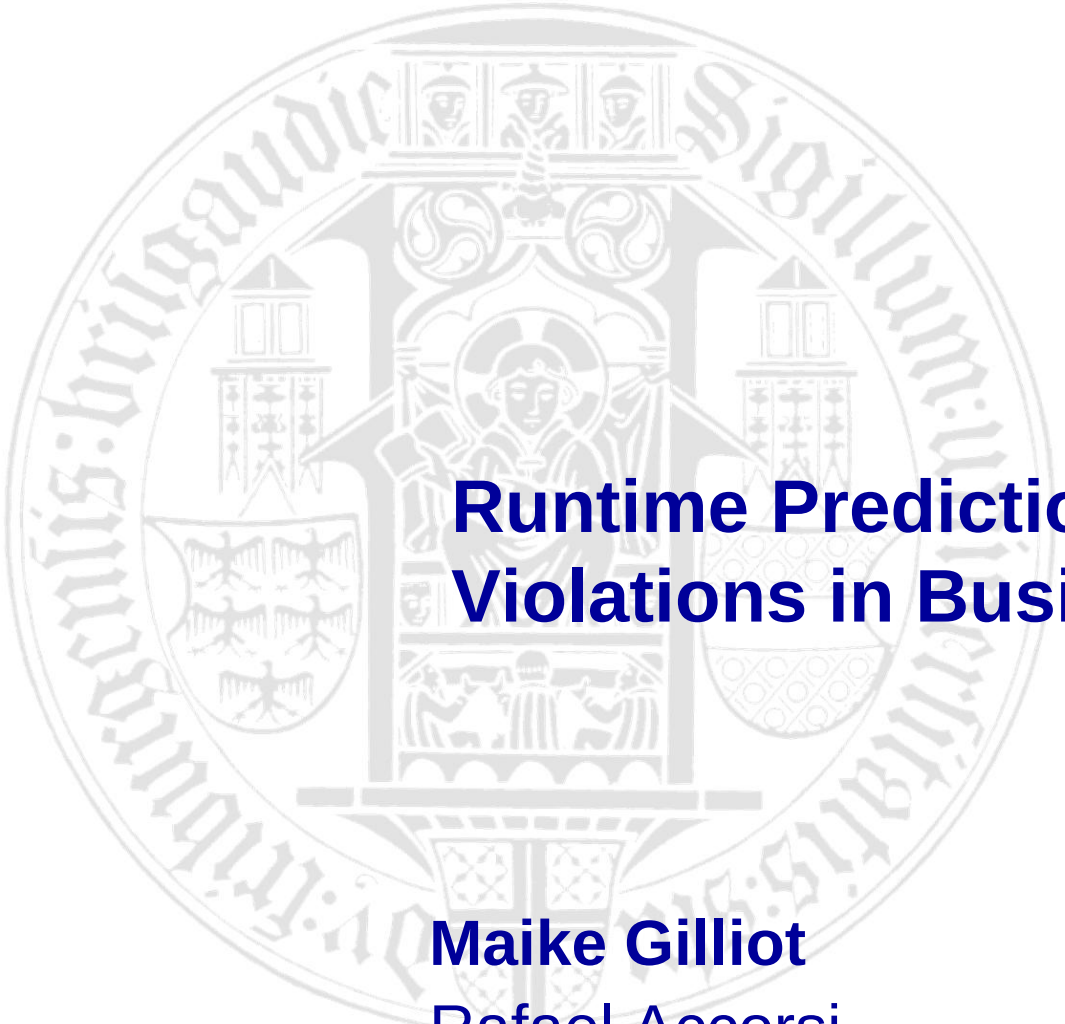


Runtime Prediction of Privacy Violations in Business Processes

Maike Gilliot
Rafael Accorsi

PRIMELife Summer School
Nice, 09th of September 2009

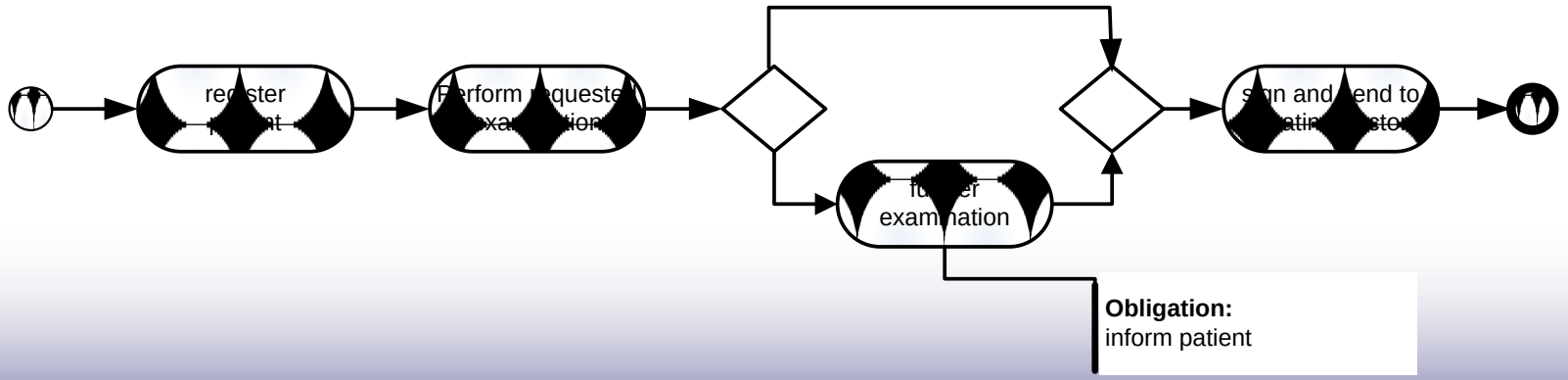
Institut of Computer Science and Social Studies, Department of Telematics
University of Freiburg

Obligations for Privacy

- **Data usage rules**
  - Prescribe usage of data **after** release
  - Examples
    - „delete data after 30 days“
    - „inform data owner about data usage“

Obligations:
actions a subject must perform

- **Obligations**
  - common in legal frameworks (up to 70%)
  - enforcement „by design“



Need for adaptive Processes

- **Need for adaptive processes**
  - Changes at runtime *to a single instance* to changed situations
  - Ability to insert or to delete steps in the workflow at runtime *to a single instance*

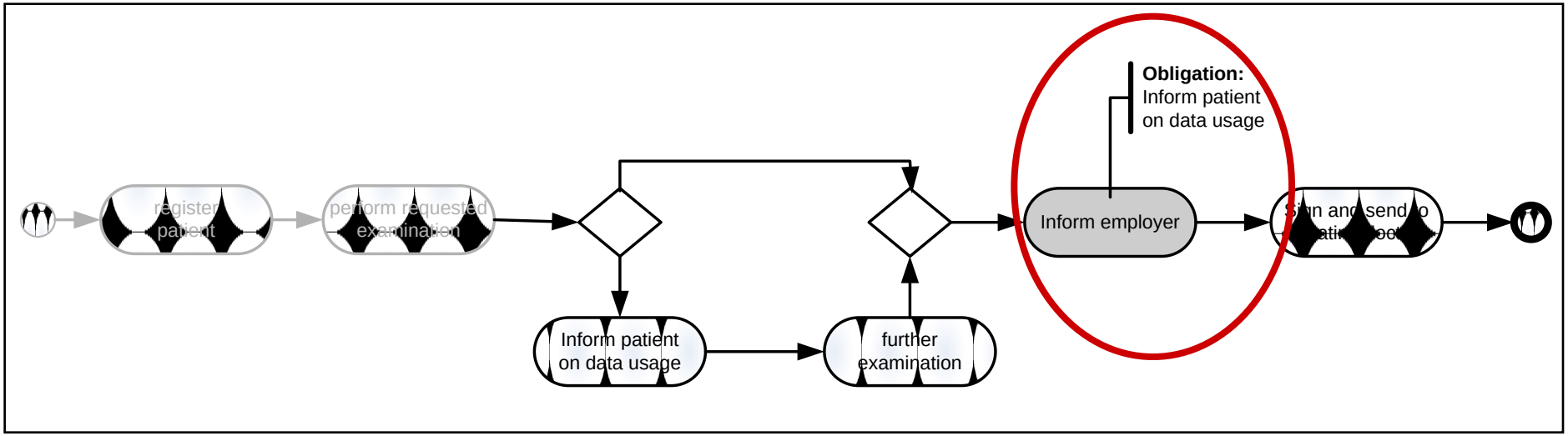
- **Assumption:**
  - Set of allowed changes for every workflow available

- **Consequence:**
  - Changed workflow may not be anymore compliant

- **Example**
  - Deletion of activity „inform patient“
  - Include activity „inform employer“

Non-compliance due to changes

In case change is accepted:



Situation:

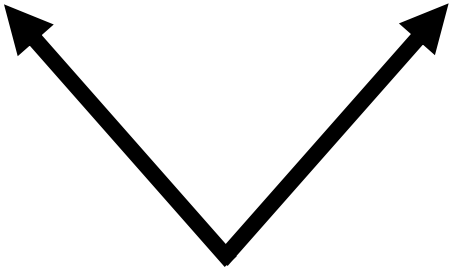
- **1 compliant continuation** {(informP),(furtherEx),(informE), (sign&send)}
- **1 non-compliant continuation** {(informE),(sign&send)}

Remedy:

- include activity „inform patient“ in upper branch
- move activity „inform patient“

Compliance in adapted processes

Problem: **Compliance** ⟷ **Adaptive processes**



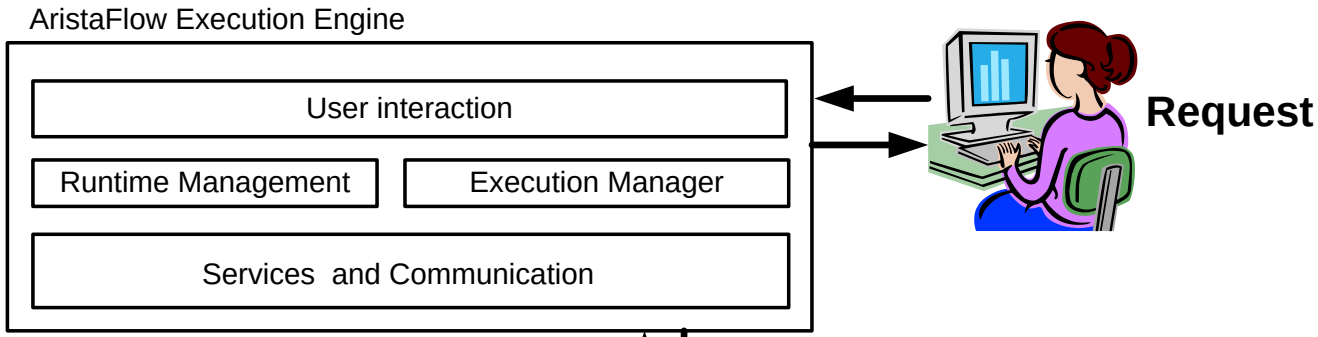
Approach: **Violation Anticipation Monitor (VAM)**

1. Evaluates change request wrt obligations
   - Detect the existence of non-compliant execution continuations
   - computes if remedy is possible
   - experience based evaluation of the continuations

2. Denies/ grants change request
   - automatically
   - delegates decision to the user

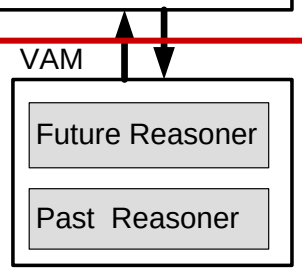
Runtime Monitoring of Change Requests

syntactic checks



evaluation wrt obligations

- **FALSE** — **deny**
- **PRESUMABLY FALSE** — **delegate to user**
- **PRESUMABLY TRUE** — **accept**
- **TRUE** — **accept**



Components of the VAM

Future Reasoner

Predicting violations based on the process continuations

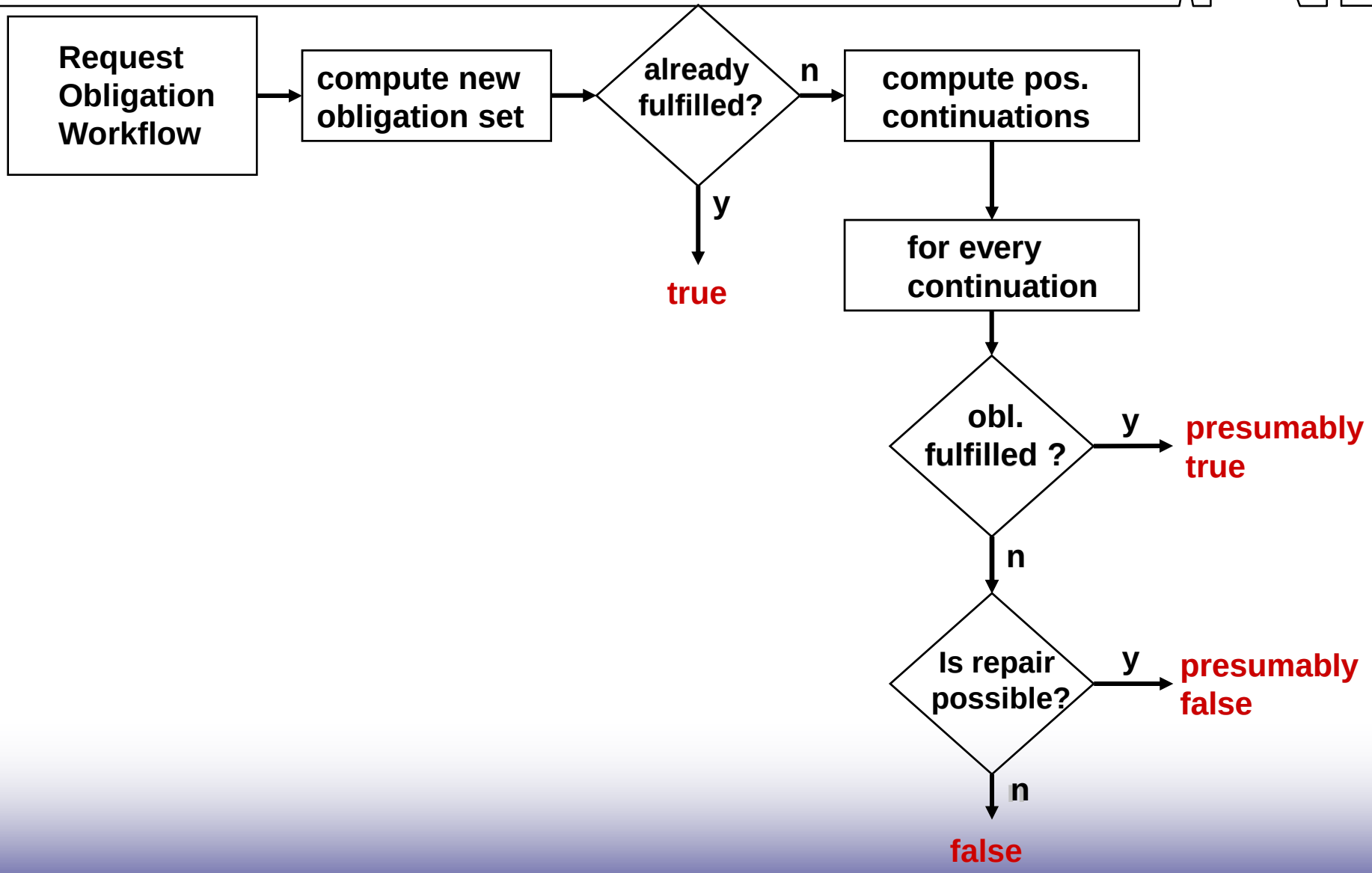
- evaltuates the possible process continuations regarding the obligation
- computes remedies

Past Reasoner

Predicting the continuation based on past executions of the workflow

- based on a log data base
- computes the probablity for each continuation

Future Reasoner



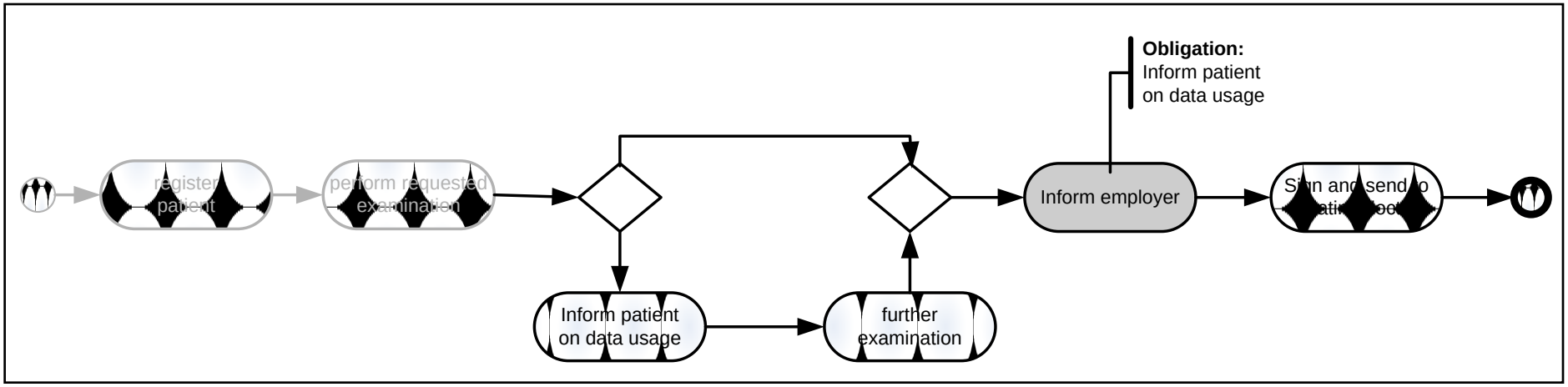
Future Reasoner - Output

Status of obligation for a single continuation

Values:

- **false:**
  the continuation violates the obligation, and no repair is possible
- **presumably false:**
  the continuation violates the obligation, but repair to the continuation is possible
- **presmuably true:**
  the continuation fulfills the obligation, if execute as it is
- **true:**
  the obligation is already fulfilled (or there is none)

Future Reasoner - Output



Situation:

- **1 compliant continuation** {(informP),(furtherEx),(informE), (sign&send)} **pt**

- **1 non-compliant continuation but repair is possible** {(informE),(sign&send)} **pf**

Output:

Continuation	$Verdict_{FR}$
(informP), (furtherEx), (informE), (sign&send)	pt
(informE), (sign&send)	pf

Past Reasoner

Log data base of former executions of the workflow

Partial trace	Continuation
…	…
(register, requestedEx)	(informP, furtherEx, informE, sign&send)
(register, requestedEx)	(sign&send)
(register, requestedEx)	(sign&send)
(register, requestedEx)	(informP, furtherEx, informE, sign&send)
(register, requestedEx)	(inform, furtherEx, informE, sign&send)
(register, requestedEx)	(inform, furtherEx, informE, sign&send)
(register, requestedEx)	(inform, furtherEx, informE, sign&send)
(register, requestedEx)	(inform, furtherEx, informE, sign&send)
…	…

Current implementation:
Number of executions per continuation

Components of the VAM

Future Reasoner

Predicting violations based on the process continuations

- evaltuates the possible process continuations regarding the obligation
- computes remedies
- **Output:**
  preliminary verdict for each continuation

Continuation	$Verdict_{FR}$
(informP), (furtherEx), (informE), (sign&send)	pt
(informE), (sign&send)	pf

Past Reasoner

Predicting the continuation based on past executions of the workflow

- based on a log data base
- computes the probablity for each continuation
- **Output:**
  probabality of each continuation

Continuation	$Prob_{PR}$
(informP), (furtherEx), (informE), (sign&send)	0.75
(informE), (sign&send)	0.25

$Verdict_{VAM}$

Future Reasoner

Continuation	$Verdict_{FR}$
(informP), (furtherEx), (informE), (sign&send)	pt
(informE), (sign&send)	pf

Past Reasoner

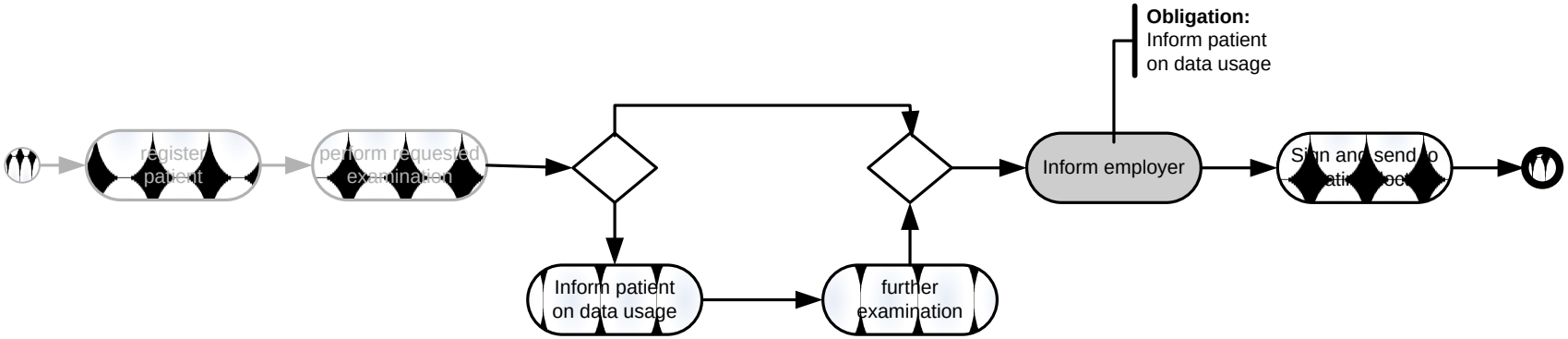
Continuation	$Prob_{PR}$
(informP), (furtherEx), (informE), (sign&send)	0.75
(informE), (sign&send)	0.25

$Verdict_{FR}$	$Prob_{PR}$
pt	0.75
pf	0.25

PRESUMABLY TRUE

Surveillance *after* change

Status of obligation may change during execution



Status:

Presumably true

Presumably false

False

Summary and Next Steps

Summary

- **Need for apdative processes *and* compliance**
- **Approach:**
  Monitoring change requests to anticipate possible violations → **VAM**
- **Result of the VAM:**

  – FALSE → reject request
  – PRESUMABLY FALSE → inform user
  – PRESUMABLY TRUE → accept request
  – TRUE → accept request

Next steps

- Implementation of the VAM
- Proof of concept evaluation
- Base Past Reasoner on Bays Nets
- Include surveillance after changes

Thank you for your attention

Maike Gilliot
gilliot@iig.uni-freiburg.de

University of Freiburg, Germany
Institut for Computer Science and Social Studies
Department of Telematics