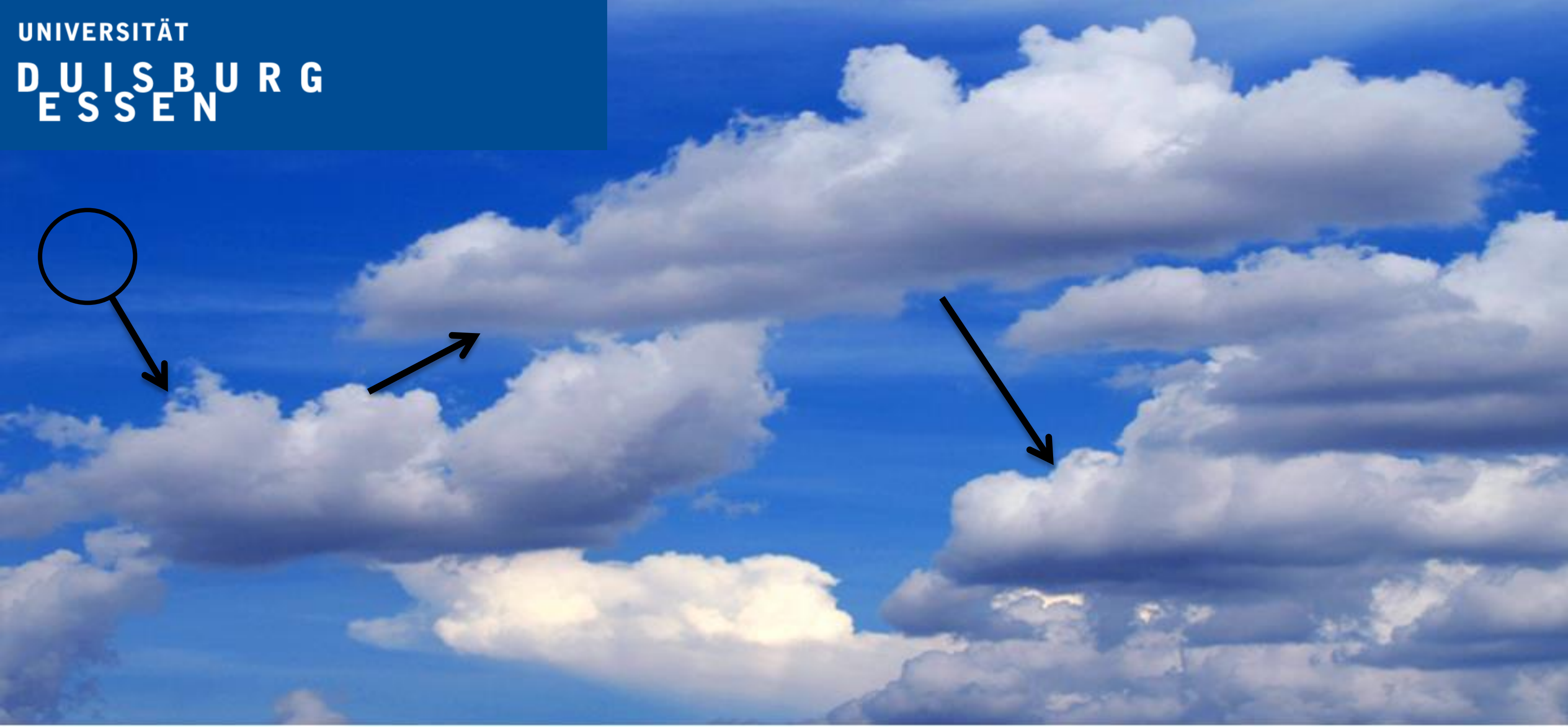


Examining Case Management Demand using Event Log Complexity Metrics

@Adaptive CM 2014 Workshop

Marian Benner-Wickner, Matthias Book, Tobias Brückmann, Volker Gruhn

PALUNO
The Ruhr Institute for Software Technology

Agenda

- Problem domain
- Event log complexity metrics
- Experimental evaluation

Problem domain

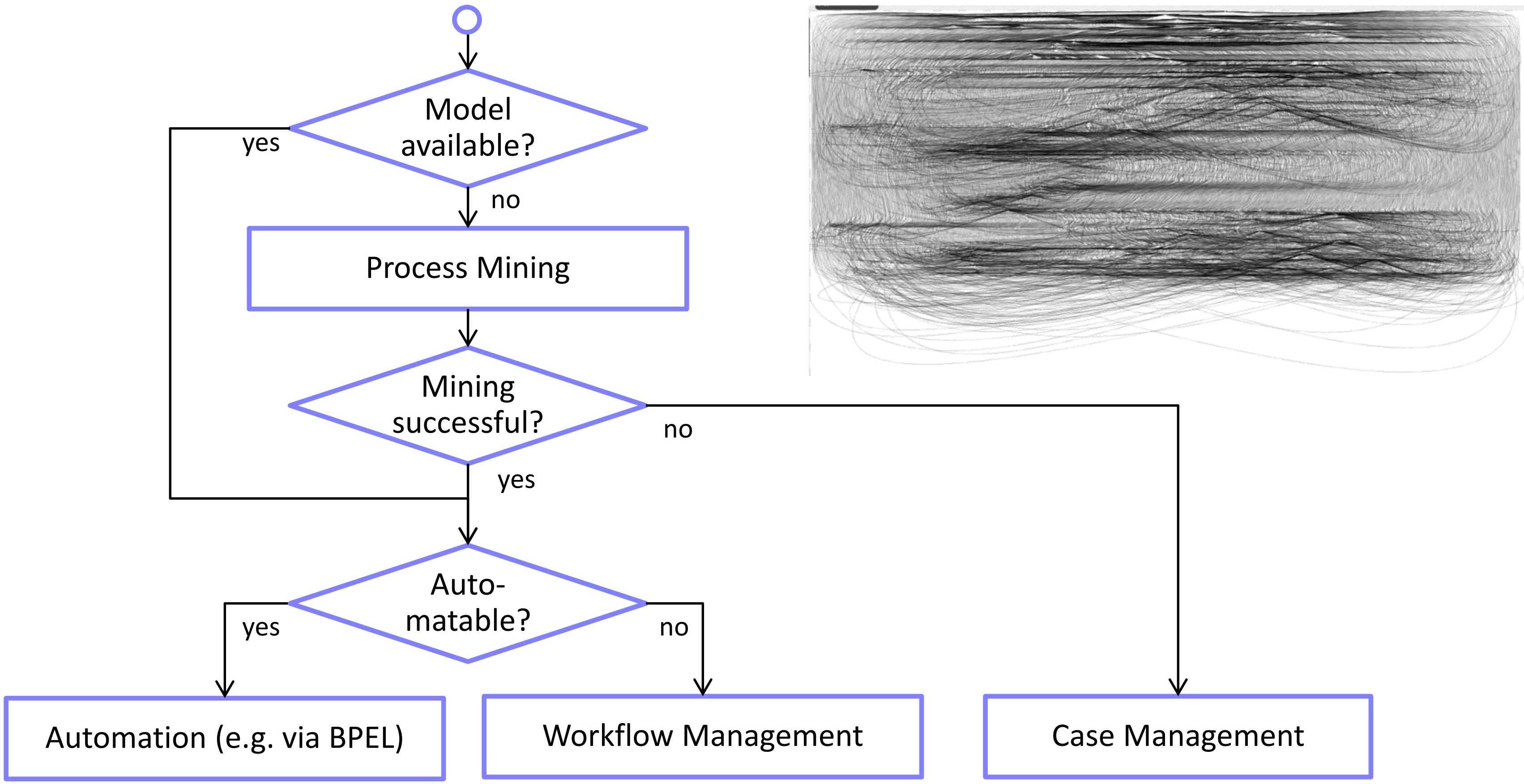
- Industrialization of structured business processes
  - State-of-the-Art: Automation and workflow support for well-known processes

	Unknown	Well-known
Automatable	Mine process, then …	Automate!
Non-automatable	Mine process, then …	Manage workflow!

- But what if…
  - …mining fails, e.g. producing “spaghetti processes”?
  - …actors permanently break/avoid workflow?
- Answer: Case Management (CM) techniques, e.g. Agenda-driven CM

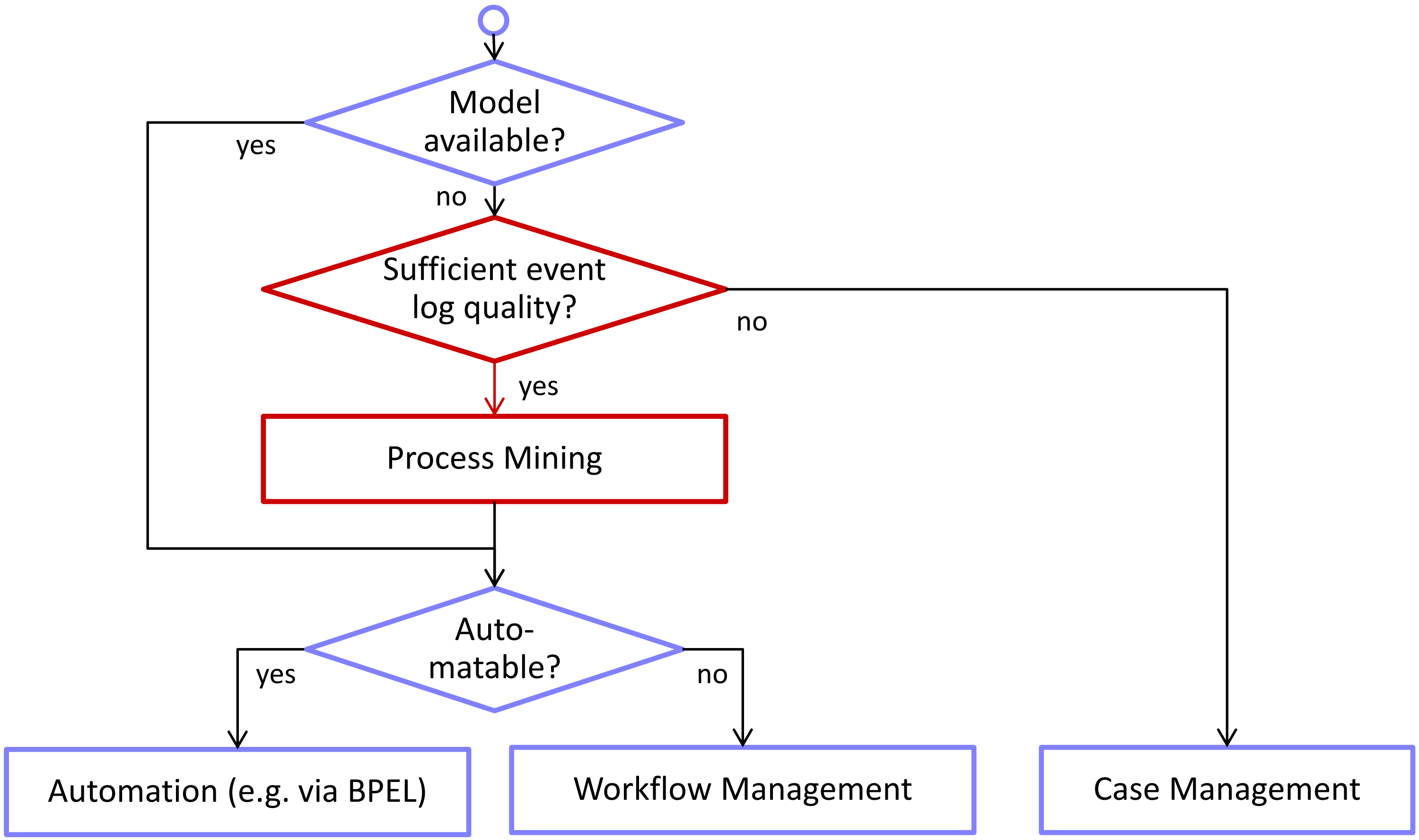
How to spot the right support technology

- Today: The "try-and-error" way



How to spot the right support technology (2)

- Suggestion of a more reasonable, convenient way:



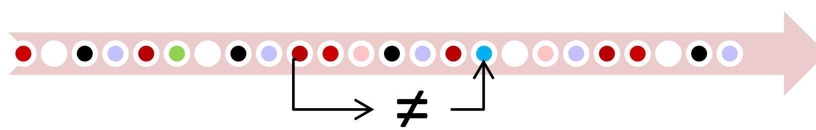
Approach

- How to measure the demand for CM techniques using event logs?
- Step 1: Define CM process characteristics
- Step 2: Introduce event log metrics!
- Step 3: Experimental evaluation

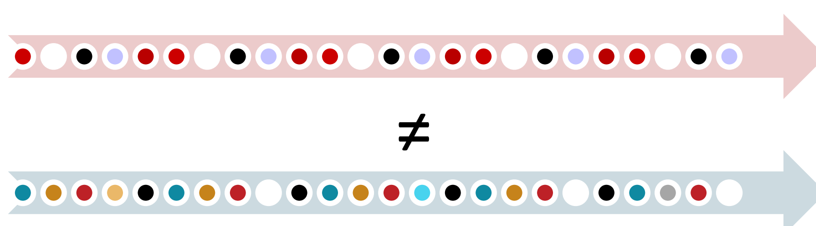
CM process characteristics

- Assumption: CM processes produce complex event logs
- Justification
  - Cases differ widely (diversity)

  ➔ Each case produces a trace (=course of events) very different to all other traces
  - Information theory lingo: a CM process event log has a high entropy
- Two dimensions of diversity
  - Event Diversity: Within any given trace, there are many different events



  - Trace Diversity: Comparing any two traces, the course of events is very different (very few re-occurring pattern)



Event log complexity metrics

Event diversity

- **Definition 1.1 (Average trace length)**
  - *k: number of traces within the event log L; k > 0.*
  - *n_i: number of each trace´s events*

$$atl(L) = \frac{1}{k}\sum_{i=1}^{k} n_i \ .$$

- Rehabilitation management example:
  - Complicated case (occupational accident)
  - Many events during various different therapies
  - Some events are re-occurring

- atl(L) does not reflect the structure

➔ Metric biased by many reoccurrences due to process loops

Event log complexity metrics

Event diversity

- **Definition 1.2 (Average trace size)**
  - *k: number of traces within the event log L*
  - *E_i : set of each trace´s distinct events*

$$ats(L) = \frac{1}{k}\sum_{i=1}^{k}|E_i|$$

- "Mean number of distinct events"
- Benefit: robust against reoccurence bias

Event log complexity metrics

Event diversity

- But what about the amount of unique events?

➔ Measure ratio between distinct events and total events

- **Definition 1.3 (Event density)**

$$ed(L) = \frac{ats(L)}{atl(L)}$$

- Consider both extremes:
  - Case A: every event is unique

    ats(L) = atl(L) => ed(L) = 1 (high density)
  - Case B: Very few events which reoccur very often

    ats(L) << atl(L) => ed(L) = 0 (low density)

Event log complexity metrics

Trace diversity

- How to quantify differences between traces of a log?
- Idea: Measure probability that a given event occurs in any trace

- **Definition 2.1 (Simple trace diversity)**.
  - *n: total number of distinct events*

$$std(L) = 1 - \frac{ats(L)}{\mathrm{n}}.$$

- Consider both extremes:
  - A few distinct events occur within each trace
    ats(L) << n => std(L) = 1 (high diversity)
  - Every distinct event occurs in every trace (at least once)
    ats(L) = n => std(L) = 0 (low diversity)

Event log complexity metrics

Trace diversity

- Yet, no metric considers the order of occurrence!
- Idea: Measure the mutual dissimilarity between all traces
- Approach:
  1. Treat traces as words and events as their characters
  2. Use levenshtein distance to measure dissimilarity

- **Definition 2.2 (Advanced trace diversity).** *Let k be the total number of traces and $LD(T_i,T_j)$ be the levenshtein distance between two traces and T_i, T_j. Recall that atl(L) is the average trace length of L. The advanced trace diversity is defined as follows:*

$$atd(L) = \frac{2}{k \times (k-1) \times atl(L)} \sum_{i=1}^{k} \sum_{j=i+1}^{k} LD(T_i, T_j)$$

Event log complexity metrics

Trace diversity

- How to calculate LD(T_i, T_j):
  1. Replace event names by a dedicated unicode character for each name
  2. Calculate levenshtein distance, treating traces as words

- Example log L_1 = {T_1, T_2, T_3, T_4, T_5} =

  {<a, b, c, d, e>, <f, g, h, i, j>, <k, l, m, n, o>, <p, q, r, s, t>, <u, v, w, x, y>}

  k = 5, atl(L_1) = 5, each LD(T_i, T_j) value is 5

$$atd(L_1) = \frac{2}{5\times(5-1)\times 5}\sum_{i=1}^{5}\sum_{j=i+1}^{5} 5 = \frac{2}{100} \times 50 = 1$$

- Example log L_2 = {T_1, T_2, T_3, T_4, T_5} =

  {<a, b, c, d, e>, <a, b, c, d, e>, <a, b, c, d, e>, <a, b, c, d, e>, <a, b, c, d, e>}

$$atd(L_2) = \frac{2}{5\times(5-1)\times 5}\sum_{i=1}^{5}\sum_{j=i+1}^{5} 0 = \frac{2}{100} \times 0 = 0$$

Experimental evaluation

Activity

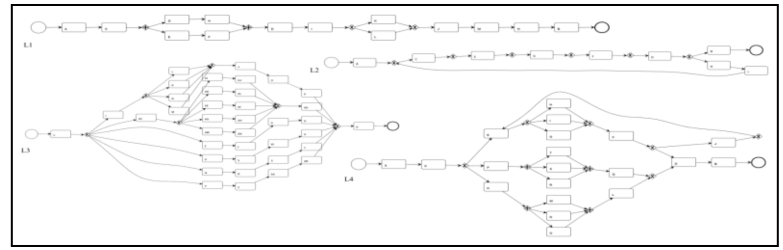
Generate artificial processes

Experiment with artificial processes

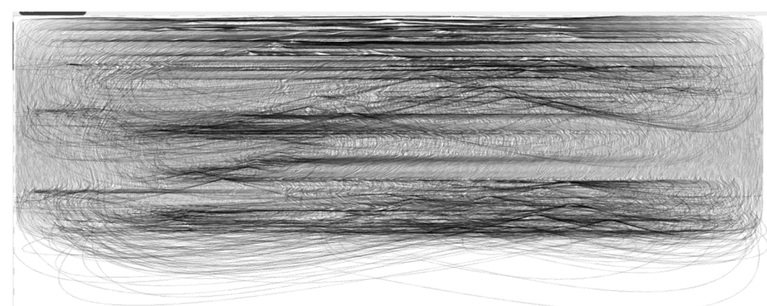
Collect real-life logs

Experiment with real-life logs

Output



➔ Metrics are valid!



➔ Metrics can indeed measure complexity in the field

Experimental evaluation

Detailed results

- Detailed results (artificial logs):

Log	atl	ats	ed (%)	std (%)	atd (%)
L_1	13	13	100	7	4
L_2	13	7	54	22	15
L_3	6	6	100	84	15
L_4	11	8	73	60	18

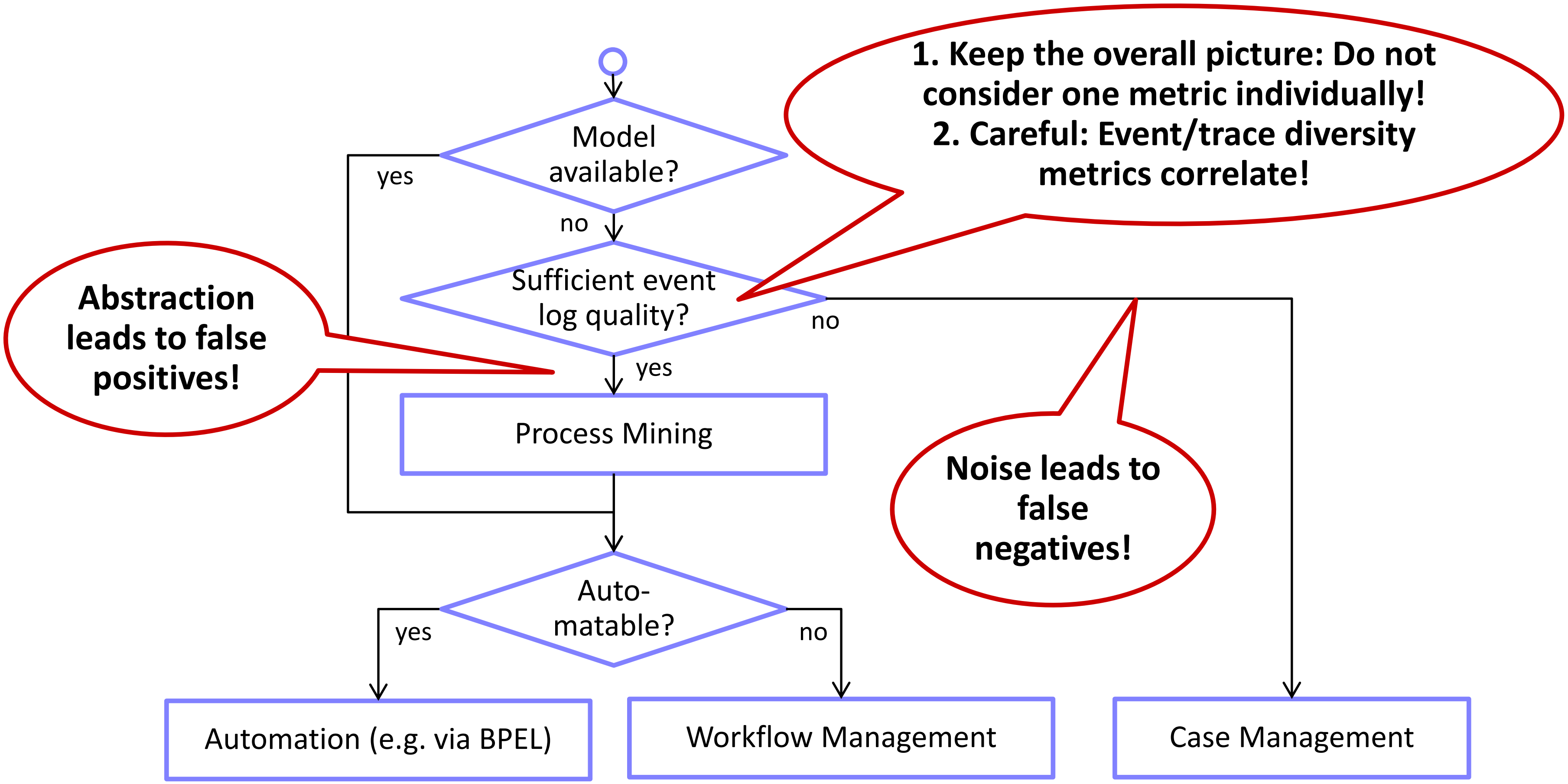
- Detailed results (real-life logs):

	Log	atl	ats	ed (%)	std (%)	atd (%)
Ad-hoc, mining impossible	L_5	131	33	25	95	37
Structured, mining possible	L_6	20	12	60	67	17
Ad-hoc (but mining successful)	L_7	9	4	44	69	19

Experimental evaluation

Detailed results

- Metrics indicate demand for alternative techniques like CM, however:



Thank you!

Questions?