Getting the Data

Barbara Re

Process Mining

Data Sources

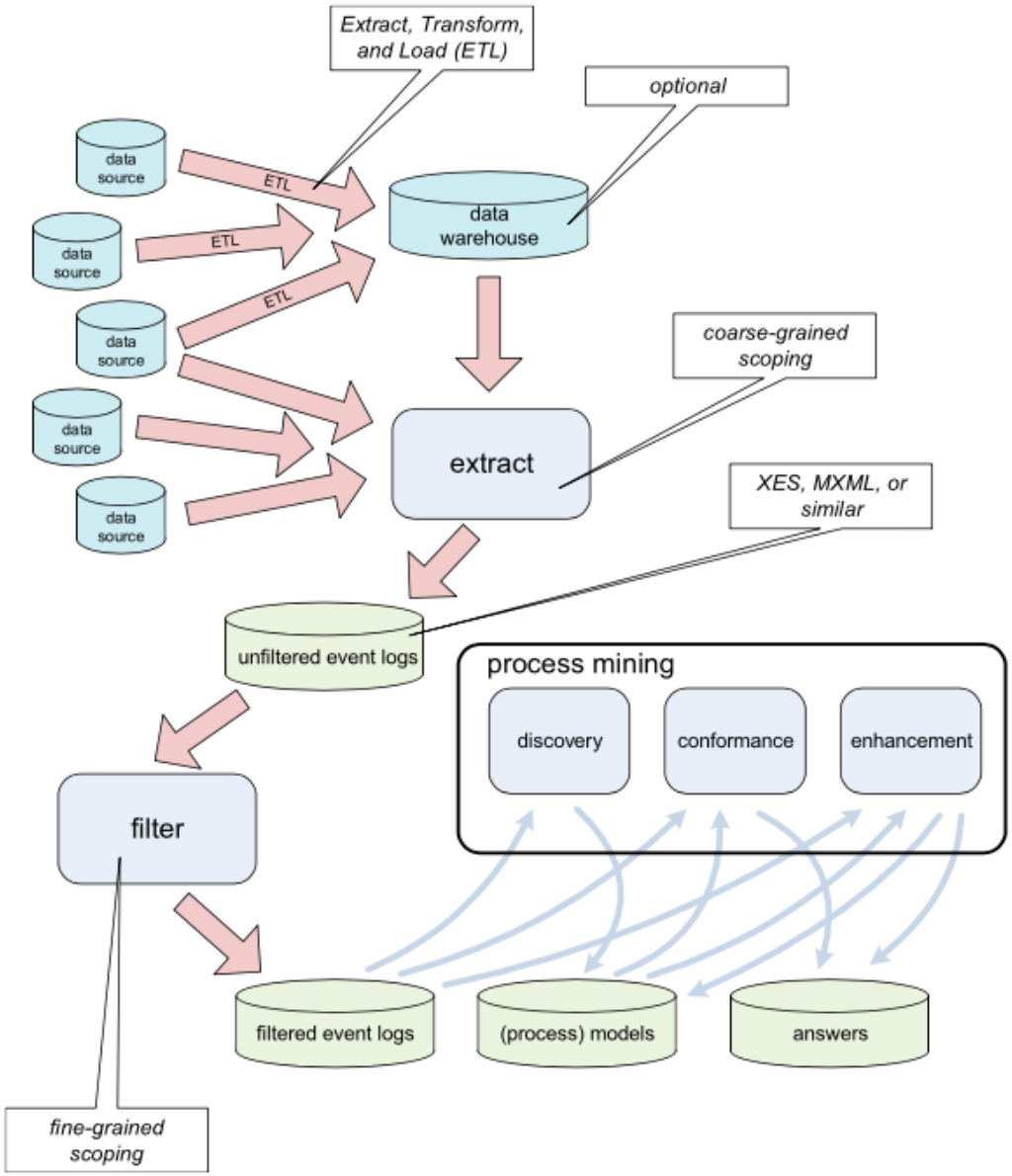
Process mining is impossible without **proper event logs**

- The challenge is to extract such data from a variety of data sources, e.g., databases, flat files, message logs, transaction logs, ERP systems, and document management systems

The goal of process mining is to answer questions about operational processes

- What really happened in the past?
- Why did it happen?
- What is likely to happen in the future?
- When and why do organizations and people deviate?
- How to control a process better?
- How to redesign a process to improve its performance?

Process Mining Workflow



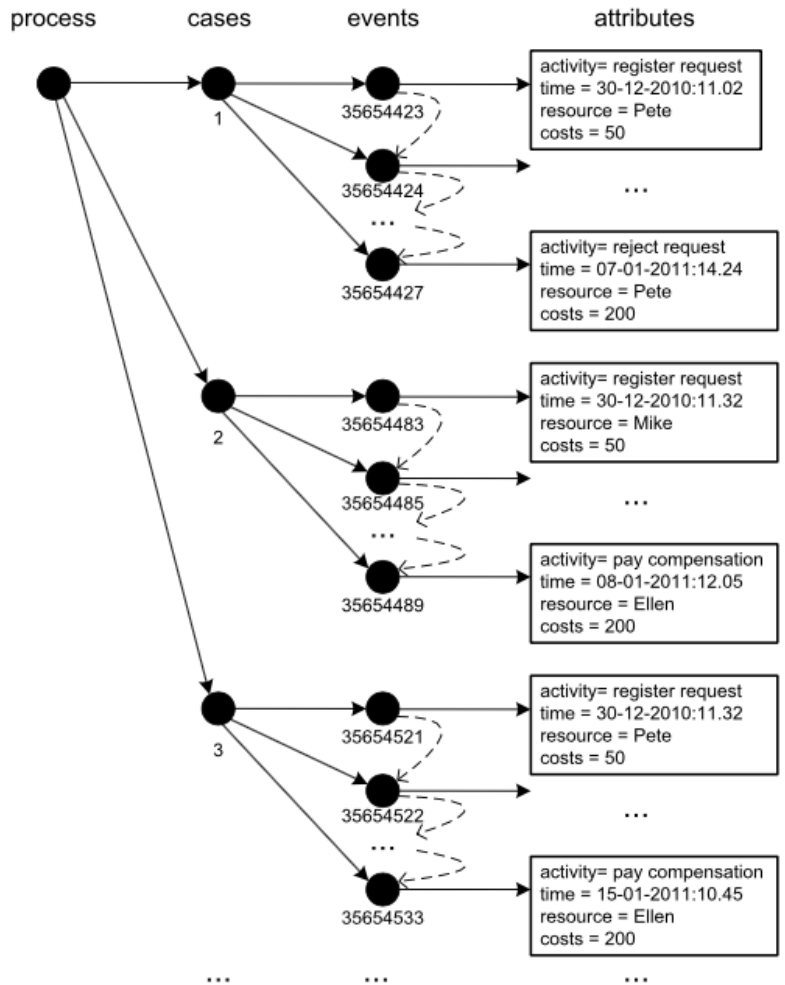
A fragment of some event log

Each line corresponds to an event

Case id	Event id	Properties				
		Timestamp	Activity	Resource	Cost	...
1	35654423	30-12-2010:11.02	register request	Pete	50	...
	35654424	31-12-2010:10.06	examine thoroughly	Sue	400	...
	35654425	05-01-2011:15.12	check ticket	Mike	100	...
	35654426	06-01-2011:11.18	decide	Sara	200	...
	35654427	07-01-2011:14.24	reject request	Pete	200	...
2	35654483	30-12-2010:11.32	register request	Mike	50	...
	35654485	30-12-2010:12.12	check ticket	Mike	100	...
	35654487	30-12-2010:14.16	examine casually	Pete	400	...
	35654488	05-01-2011:11.22	decide	Sara	200	...
	35654489	08-01-2011:12.05	pay compensation	Ellen	200	...
3	35654521	30-12-2010:14.32	register request	Pete	50	...
	35654522	30-12-2010:15.06	examine casually	Mike	400	...
	35654524	30-12-2010:16.34	check ticket	Ellen	100	...
	35654525	06-01-2011:09.18	decide	Sara	200	...
	35654526	06-01-2011:12.18	reinitiate request	Sara	200	...
	35654527	06-01-2011:13.06	examine thoroughly	Sean	400	...
	35654530	08-01-2011:11.43	check ticket	Pete	100	...
	35654531	09-01-2011:09.55	decide	Sara	200	...
	35654533	15-01-2011:10.45	pay compensation	Ellen	200	...
4	35654641	06-01-2011:15.02	register request	Pete	50	...
	35654643	07-01-2011:12.06	check ticket	Mike	100	...
	35654644	08-01-2011:14.43	examine thoroughly	Sean	400	...
	35654645	09-01-2011:12.02	decide	Sara	200	...
	35654647	12-01-2011:15.44	reject request	Ellen	200	...
...	...	...	...	...	...	...

Structure of event logs

- A process consists of cases (also called instance)
- A case consists of events such that each event relates to precisely one case
- Events within a case are ordered
- Events can have attributes (e.g. of typical attribute names are activity, time, costs, and resource)



Structure of event logs

Definition (Event)

Let $\mathcal{E}$ be the `event universe`, i.e., the set of all possible event identifiers

Events may be characterized by various `attributes`, e.g., an event may have a timestamp, correspond to an activity, is executed by a person, has associated costs, etc.

Definition (Attribute)

Let $\mathcal{AN}$ be a set of attribute names. For any `event` $e \in \mathcal{E}$ and `name` $n \in \mathcal{AN}$:

- $\#n(e)$ is the value of attribute n for event e
- If event e does not have an attribute named n, then $\#n(e) = \perp$ (null value)

Standard attributes we will assume some conventions

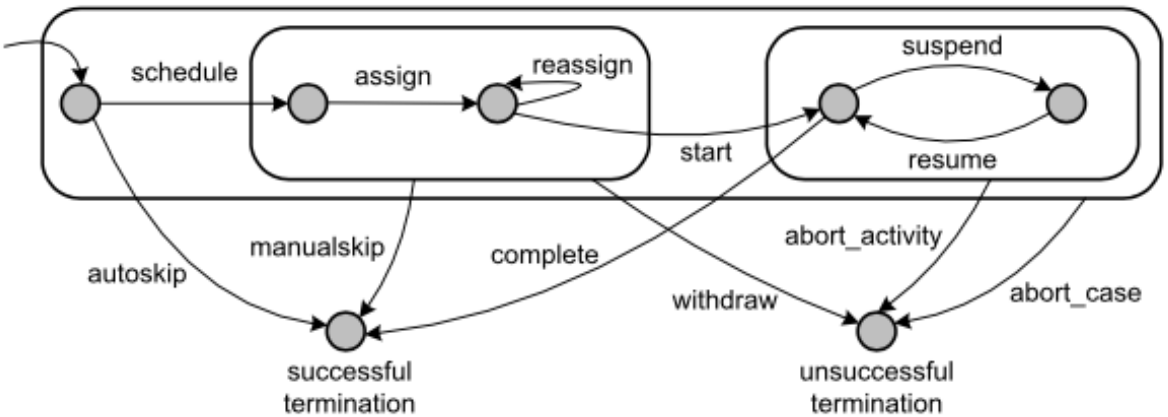
For convenience we assume the following standard attributes (none of these attributes is mandatory):

- $\#activity(e)$ is the activity associated to event e
- $\#time(e)$ is the timestamp of event e
- $\#resource(e)$ is the resource associated to event e
- $\#trans(e)$ is the transaction type associated to event e, examples are schedule, start, complete, and suspend

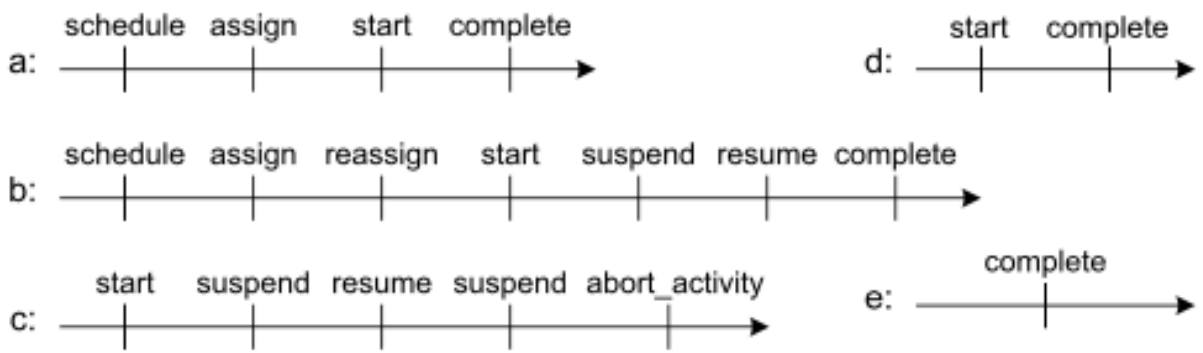
Timestamps should be **non-descending** in the event log
We assume a time domain $\mathcal{T}$, i.e., $\#time(e) \in \mathcal{T}$ for any $e \in \mathcal{E}$

Transaction type attribute

The transaction type attribute $\#trans(e)$ refers to the life-cycle of activities, we assume the following transactional life-cycle model



Example: transactional events for five different activity instances



Overlapping activity instances

We often refer to the event by its activity name - Technically this is not correct!!!!

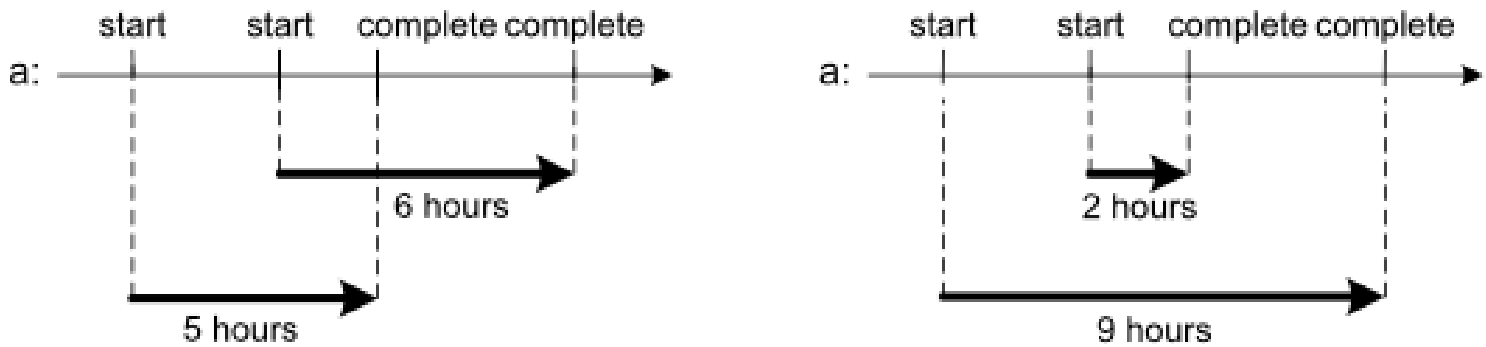
There may be many **events** that refer to the same **activity name**, within a case these events may refer:

- to the **same activity instance** (e.g., start and complete events)
- to **different activity instances** (e.g., in a loop)

This distinction is very important when measuring service times, waiting times, etc.

Example: Consider the scenario in which **the same activity is started twice** for the same case, i.e., two activity instances are running in parallel, and then one of them completes.

Did the activity that was started first complete or the second one?



They leave the same footprint in the event log!!!

Solving Correlation Problems

- by adding information to the log
  - When implementing systems, such information can easily be added to the logs; just provide an activity instance attribute to keep track of this
  - When dealing with existing systems this is not as simple as it seems, for example, when correlating messages between organizations there may be the need to scan the content of the message to find a suitable identifier (e.g., address or name)
- by using heuristics
  - One could just assume a first-in-first-out order and pick the first scenario!
  - One may introduce timeouts when the time between a start event and complete event is too long

Classifier

Some process mining techniques take into account the **transactional model** whereas others just consider **atomic events**

Sometimes we just want to focus:

- on complete events
- on withdrawals

This can be supported by filtering (e.g., removing events of a particular type) and by the concept of a **classifier**

A classifier is a function that maps the attributes of an event onto a label used in the resulting process model

Definition (Classifier)

For any event $e \in \mathcal{E}$ `event universe`, $\underline{e}$ is the name of the event

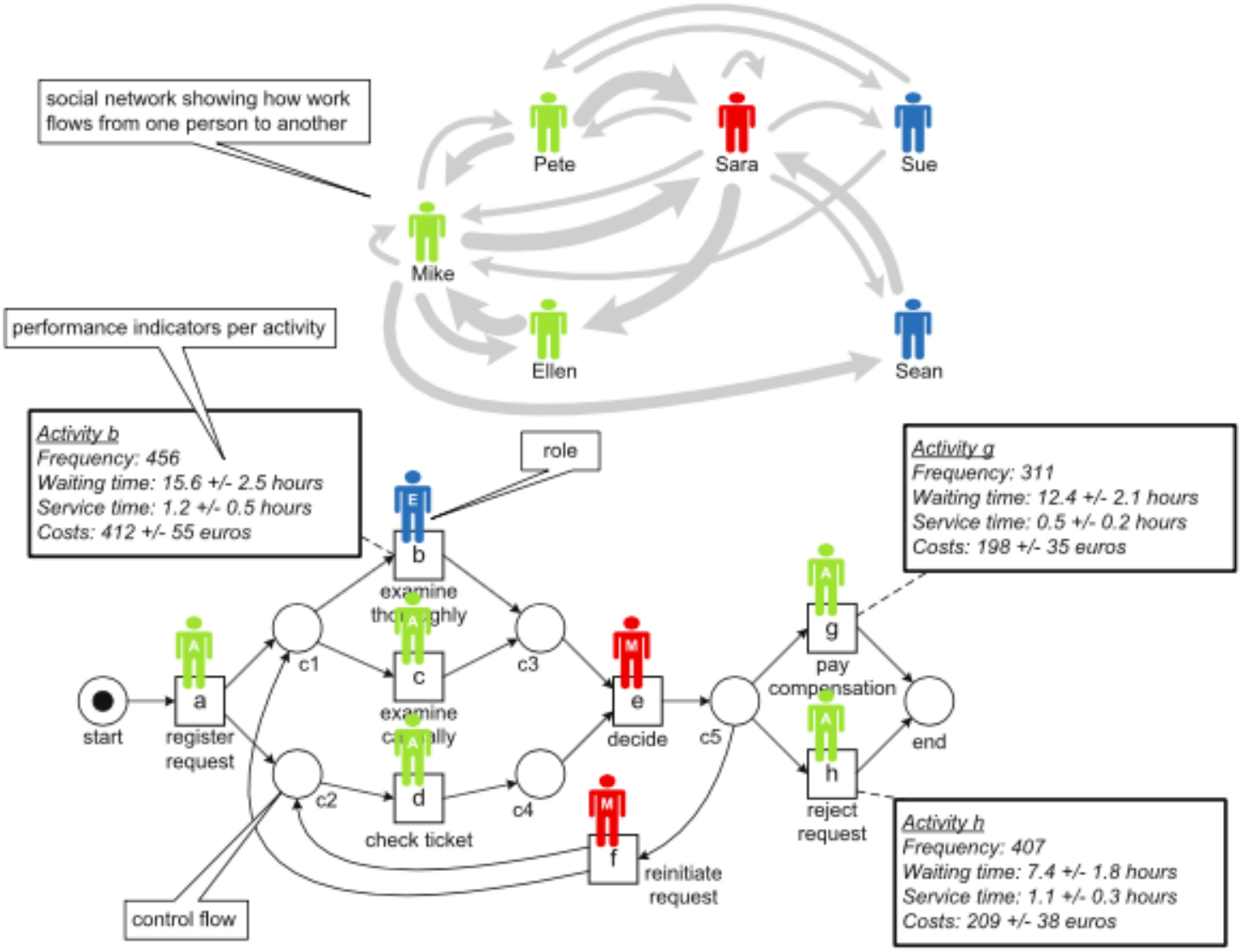
Classifier

In case $\underline{e} = \#activity(e)$ -> $\langle (a, b, c, d, e) \rangle$

In case $\underline{e} = \#activity(e), \#trans(e)$ -> $\langle (a, schedule)(a, assign)(a, start)(a, complete) \rangle$

We assume the classifier $\underline{e} = \#activity(e)$ as the default classifier
This is why we considered the activity attribute to be mandatory

Process Mining Types



XES - eXtensible Event Stream

- Until 2010 the de facto standard for storing and exchanging event logs was MXML (Mining eXtensible Markup Language)
- In September 2010, the XES format was adopted by the **IEEE Task Force on Process Mining** and became the **de facto exchange format for process mining**
- On November 11th, 2016, the XES Standard has been officially published by the IEEE as an **official IEEE standard**
- See **www.xes-standard.org** for detailed information about the standard

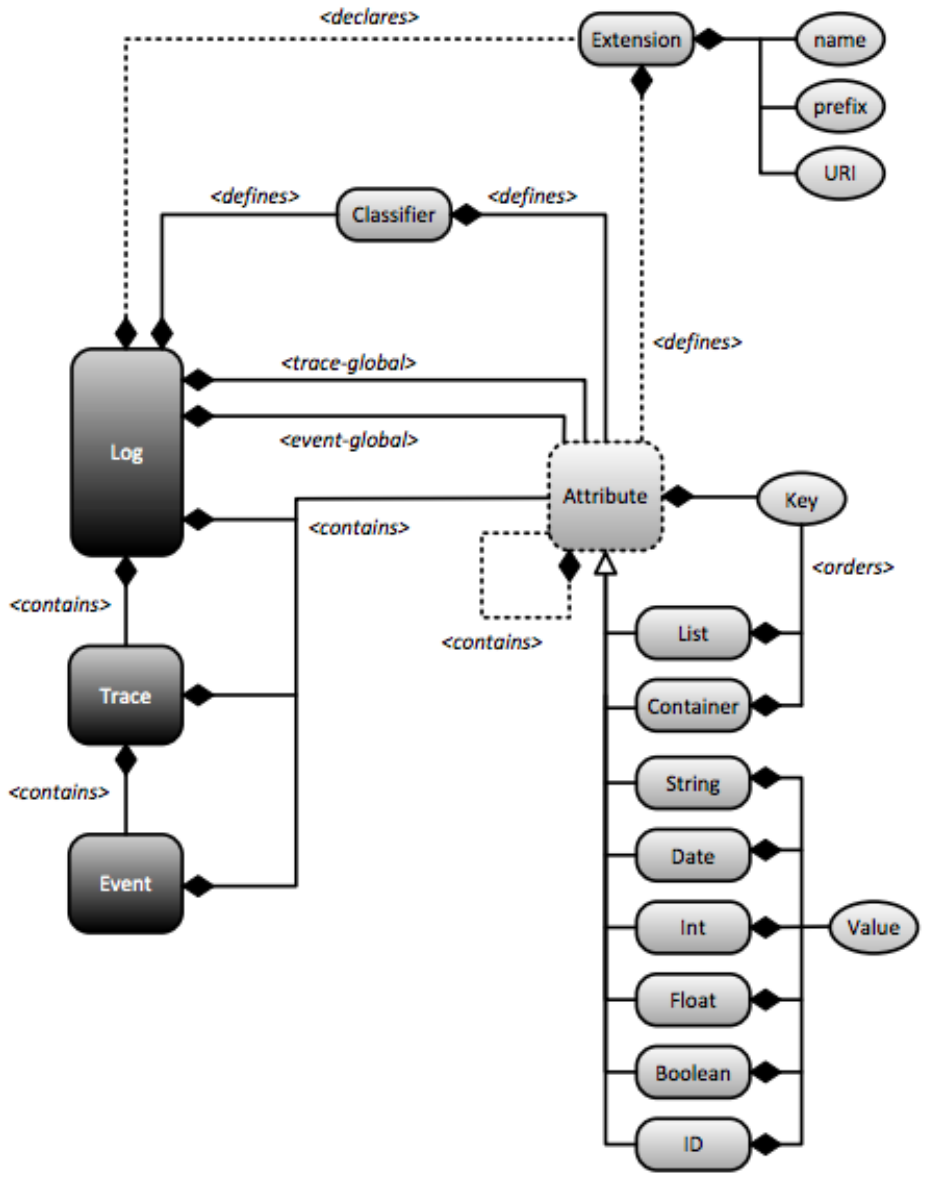
XES - eXtensible Event Stream

- The XES standard defines a grammar for a tag-based language whose aim is to provide designers of information systems with a unified and extensible methodology for capturing systems behaviors by means of event logs and event streams is defined in the XES standard
- An XML Schema describing the structure of an XES event log/stream and a XML Schema describing the structure of an extension of such a log/stream are included in this standard
- A basic collection of so-called XES extension prototypes that provide semantics to certain attributes as recorded in the event log/stream is included in this standard

XES - Principles

- **Simplicity** Use the simplest possible way to represent information. XES logs should be easy to parse and to generate, and they should be equally well human-readable. In designing this standard, care has been taken to take a pragmatic route wherever that benefits an ease of implementation.
- **Flexibility** The XES standard should be able to capture event logs from any background, no matter what the application domain or IT support of the observed process. Thus, XES aims to look beyond process mining and business processes, and strives to be a general standard for event log data.
- **Extensibility** It must be easy to add to the standard in the future. Extension of the standard should be as transparent as possible, while maintaining backward and forward compatibility. In the same vein, it must be possible to extend the standard for special requirements, e.g. for specific application domains, or for specific tool implementations.
- **Expressivity** While striving for a generic format, event logs serialized in XES should encounter as little loss of information as possible. Thus, all information elements must be strongly typed, and there must be a generic method to attach human-interpretable semantics to them.

XES meta model expressed in terms of a UML class diagram



XES meta model expressed in terms of a UML class diagram

- Log object, which contains all event information that is related to one specific process
- A log contains an arbitrary number (may be empty) of trace objects. Each trace describes the execution of one specific instance, or case, of the logged process
- Every trace contains an arbitrary number (may be empty) of event objects. Events represent atomic granules of activity that have been observed during the execution of a process.
- The log, trace, and event objects contain no information themselves. They only define the structure of the document. All information in an event log is stored in attributes. Logs, traces, and events each contain an arbitrary number of attributes. There are six types of elementary attributes, each defined by the type of data value they represent. Next to these elementary attributes, there are two types of collection attributes. For providing maximum flexibility in data storage, XES allows nested attributes, i.e. attributes can themselves have child attributes (note that this feature is required when using lists and/or containers).

XES classifier

The XES format makes event classification configurable and flexible, by introducing the concept of event classifiers. An event classifier assigns to each event an identity, which makes it comparable to other events (via their assigned identity).

Classifiers are defined via a set of attributes, from which the class identity of an event is derived. In its simplest form, an event classifier is defined by one attribute, and the value of that attribute would yield the class identity of an event.

< classifier name="Activity classifier" keys="name status" / >

Challenges when extracting event logs

```
<?xml version="1.0" encoding="UTF-8" ?>
<log xes.version="2.0" xes.features="arbitrary-depth" xmlns="http://www.xes-standard.org
    /">
    <extension name="Concept" prefix="concept" uri="http://www.xes-standard.org/concept.
        xesext"/>
    <extension name="Time" prefix="time" uri="http://www.xes-standard.org/time.xesext"/>
    <global scope="trace">
        <string key="concept:name" value=""/>
    </global>
    <global scope="event">
        <string key="concept:name" value=""/>
        <date key="time:timestamp" value="1970-01-01T00:00:00.000+00:00"/>
        <string key="system" value=""/>
    </global>
    <classifier name="Activity" keys="concept:name"/>
    <classifier name="Another" keys="concept:name system"/>
    <float key="log attribute" value="2335.23"/>
    <trace>
        <string key="concept:name" value="Trace number one"/>
        <event>
            <string key="concept:name" value="Register client"/>
            <string key="system" value="alpha"/>
            <date key="time:timestamp" value="2009-11-25T14:12:45:000+02:00"/>
            <int key="attempt" value="23">
                <boolean key="tried hard" value="false"/>
            </int>
        </event>
        <event>
            <string key="concept:name" value="Mail rejection"/>
            <string key="system" value="beta"/>
            <date key="time:timestamp" value="2009-11-28T11:18:45:000+02:00"/>
        </event>
    </trace>
</log>
```

Challenges when extracting event logs

- **Correlation** - Events in an event log are grouped per case. This simple requirement can be quite challenging as it requires event correlation, i.e., events need to be related to each other.
- **Timestamps** - Events need to be ordered per case. Typical problems: only dates, different clocks, delayed logging.
- **Snapshots** - Cases may have a lifetime extending beyond the recorded period, e.g., a case was started before the beginning of the event log.
- **Scoping** - How to decide which tables to incorporate?
- **Granularity** - The events in the event log are at a different level of granularity than the activities relevant for end users.