XES

Reflections and Recommendations

prof.dr.ir. Wil van der Aalst
RWTH Aachen University
W: vdaalst.com T:@wvdaalst
Adopted as an IEEE standard in 2016, but this started in 2009 with a first draft of XES as input for the TFPM.

From the Minutes of the Meeting of the IEEE Task Force on Process Mining, September 15th 2010, Stevens Institute of Technology, Hoboken NJ: “... All participants agree with using the format in the future. ... The task force adopts the XES standard and will use it as a tool to promote the topic.”

Participants
At the time this IEEE standard was completed, the XES Working Group had the following membership:

Wil van der Aalst, Chair
Christian Günther, Vice Chair

The following members of the individual balloting committee voted on this standard. Balloters may have voted for approval, disapproval, or abstention.

- Giovanni Acampora
- Michael Bauer
- JC Bose
- Keith Chow
- Randall Groves
- Christian Günther
- Antonella Guzzo

- Werner Hitzl
- Noriyuki Kanauchi
- Piotr Karocki
- Akhil Kumar
- Edward McCall
- Björn Molitor
- Takahide Nagayama

- Zhigiew Paszkiewicz
- Michal Rosik
- Christopher Turner
- Wil van der Aalst
- Eric Verbeek
- Lijie Wen
- Oren Yuen
Focus on XES syntax was unintended

In hindsight a few things were probably over-engineered.

See also alternative XES storage formats, e.g., JXES (JSON-based XES).
Example realization in Celonis
We need more “event producers”

- Why are not all vendors supporting this although it is 2 days of work?
  - End-users are not demanding this.
  - Most XES data sets are for scientific competitions.
- Game will change when ERP/CRM vendors start to export event data in XES format.
  - How to make that happen?
Activity instances: Important, but rarely used (not logged and not used in tools)

7.1 Concept extension
The Concept extension defines, for all levels of the XES type hierarchy, an attribute which stores the generally understood name of type hierarchy elements.

<table>
<thead>
<tr>
<th>Name</th>
<th>Prefix</th>
<th>URI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Concept</td>
<td>concept</td>
<td><a href="http://www.xes-standard.org/concept.xesext">http://www.xes-standard.org/concept.xesext</a></td>
</tr>
</tbody>
</table>

7.1.1 Attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Key</th>
<th>Components</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instance</td>
<td>instance</td>
<td>Event</td>
<td>xx:string</td>
<td>This represents an identifier of the activity instance whose execution has generated the event. This way, multiple instances (occurrences) of the same activity can be told apart.</td>
</tr>
</tbody>
</table>

Instance: “This represents an identifier of the activity instance whose execution has generated the event. This way, multiple instances (occurrences) of the same activity can be told apart.”
Cost Extension: Also rarely used (not logged and not used in tools)

### 7.7 Cost extension

The cost extension defines a nested element to store information about the cost associated with activities within a log. The objective of this extension is to provide semantics to cost aspects that can be associated with events in a log. The definition associates three data elements with a particular cost element: the amount associated with the cost element as well as the cost driver that is responsible for incurring that cost and the cost type. As it is possible for more than one cost element to be associated with an event, the cost incurred per event is summarized using the total attribute. The currency element is also recorded once per event. Cost information can be recorded at the trace level (for instance, to be able to say that it costs $20 when a case is started). Cost information can also be recorded at the event level (for instance, for certain event types such as complete or canceled events) to capture the cost incurred in undertaking the activity by a resource.

<table>
<thead>
<tr>
<th>Name</th>
<th>Key</th>
<th>Components</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amount</td>
<td>amount</td>
<td>Meta</td>
<td>xs:double</td>
<td>The value contains the cost amount for a cost driver.</td>
</tr>
<tr>
<td>Driver</td>
<td>driver</td>
<td>Meta</td>
<td>xs:string</td>
<td>The value contains the id for the cost driver.</td>
</tr>
<tr>
<td>Type</td>
<td>type</td>
<td>Meta</td>
<td>xs:string</td>
<td>The value contains the cost type (e.g., Fixed, Overhead, Materials).</td>
</tr>
</tbody>
</table>

The drivers attribute shall contain any number of driver attributes, and every driver attribute shall contain the amount and type attribute, like follows:

```xml
<event>
  <string key="cost:currency" value="AUD"/>
  <string key="cost:total" value="123.50"/>
  <list key="cost:drivers">
    <values>
      <string key="driver" value="d2f4ee27">
        <float key="amount" value="21.40"/>
        <string key="type" value="Labour"/>
      </string>
      <string key="driver" value="abc124">
        <float key="amount" value="102.10"/>
        <string key="type" value="Variable Overhead"/>
      </string>
    </values>
  </list>
</event>
```

### 7.7.1 Attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Key</th>
<th>Components</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total</td>
<td>total</td>
<td>Trace, Event</td>
<td>xs:double</td>
<td>Total cost incurred for a trace or an event. The value represents the sum of all the cost amounts within the element.</td>
</tr>
<tr>
<td>Currency</td>
<td>currency</td>
<td>Trace, Event</td>
<td>xs:string</td>
<td>The currency (using the ISO 4217:2008 standard) of all costs of this element.</td>
</tr>
<tr>
<td>Drivers</td>
<td>drivers</td>
<td>Trace, Event, List</td>
<td></td>
<td>A detailed list containing cost driver details.</td>
</tr>
</tbody>
</table>
Each event refers to a single case and events are only related through cases!
### Event Data Representation

**event = activity + timestamp + objects + attributes**

See Wil van der Aalst: Object-Centric Process Mining: Dealing with Divergence and Convergence in Event Data. SEFM 2019, 3-25 https://doi.org/10.1007/978-3-030-30446-1_1
Convergence and Divergence

- If item is the case id, then activities at the order and package level get replicated (convergence problem).
  - \( \text{place\_order}(\text{order1}, \text{item1}, \text{item2}, \text{item3}) \) becomes \( \text{place\_order}(\text{item1}), \text{place\_order}(\text{item2}), \text{place\_order}(\text{item3}) \)

- If order is the case id, then causal relations at the level of items get lost (divergence problem).
  - \( \text{pick\_item}(\text{order1}, \text{item1}), \text{pick\_item}(\text{order1}, \text{item2}), \text{pack\_item}(\text{order1}, \text{item1}), \text{pack\_item}(\text{order1}, \text{item2}), \) becomes \( \text{pick\_item}(\text{order1}), \text{pick\_item}(\text{order1}), \text{pack\_item}(\text{order1}), \text{pack\_item}(\text{order1}) \)
The purpose of the OCEL standard is to provide a general standard to interchange object-centric event data with multiple case notions. We set the following goals for the standard:

- **Interoperability**: with the provision of the OCEL standard and JSON/XML serializations of OCEL, we want to support a widespread collection of languages and systems.
- **Generalization**: the standard supports the storage of events, objects, and their attributes. Furthermore, the standard can be extended.
- **Provision of a collection of examples**: example logs, extracted from information systems supporting some widespread business processes, are provided for the OCEL standard.
- **Tool/Library Support**: to support the implementation of OCEL in custom applications, tool/library support shall be provided.

```xml
<events>
  <event>
    <string key="id" value="e1"/>
    <string key="activity" value="place_order"/>
    <date key="timestamp" value="2020-07-09T08:20:01.527+01:00"/>
    <list key="omap">
      <string key="object-id" value="i1"/>
      <string key="object-id" value="o1"/>
      <string key="object-id" value="i2"/>
    </list>
    <list key="vmap">
      <string key="resource" value="Alessandro"/>
      <float key="prepaid-amount" value="200.0"/>
    </list>
  </event>
  <event>
    <string key="id" value="e2"/>
    <string key="activity" value="check_availability"/>
    <date key="timestamp" value="2020-07-09T08:21:01.527+01:00"/>
  </event>
</events>
```

http://ocel-standard.org/
Conclusion

Lessons Learned
Lessons Learned

• Keep it **simple**, but not too simple.
• Learn to crawl before you can walk (i.e., **use XES first**).
• Be clear about the **core concepts**, e.g., is an event atomic or not (if so, you need activity instances).
• Need to be able to **relate events** in a **flexible manner**.
• Need to be able to **relate events and objects** (see ▲).
• We need **domain specific ontologies** instead of standard extensions, e.g., the P2P and O2C ontology.
• **Concepts first**: Multiple exchange formats.