Open Event Expression Language

Engineering Session

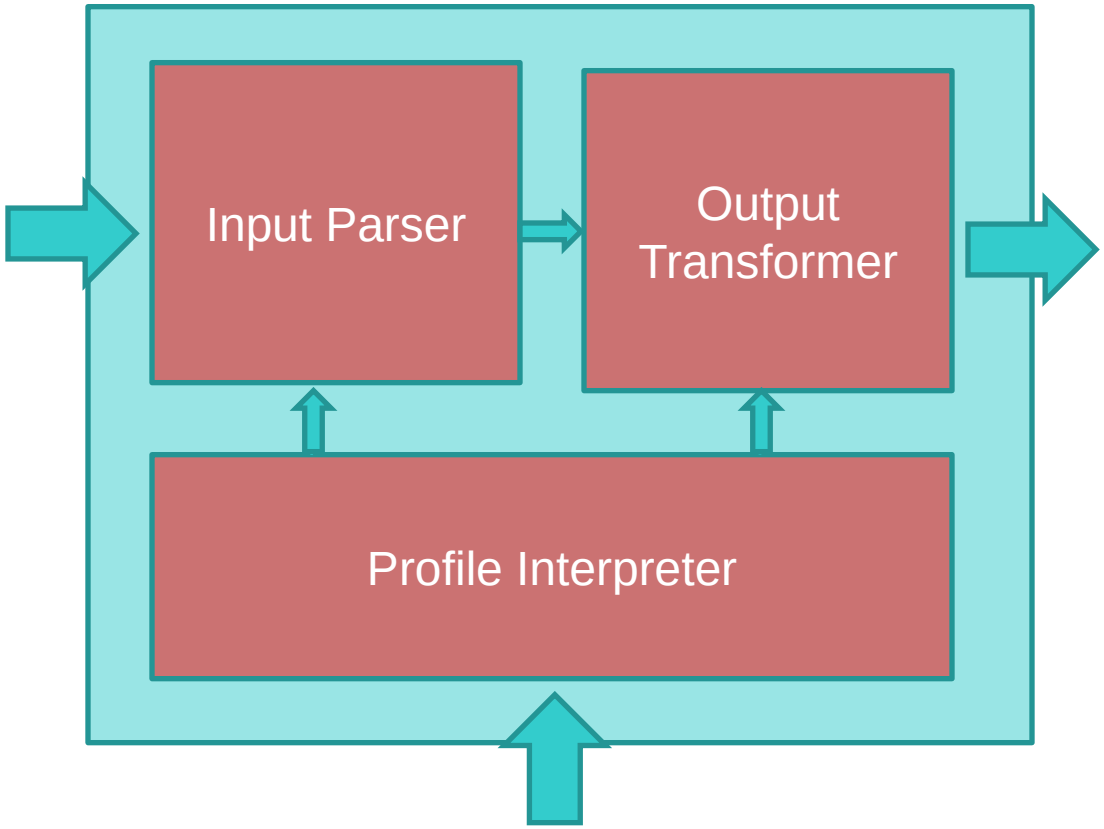
OEEL

- Vision for the specification
- Notional Architecture
- Data Transformation
- Examples
- Flexibility
- Issues
- Impact on Vendors
- Content Transformation

OEEL Vision

- Provide a standardized ability to represent parsing logic external to the parsing application
  - Provide vendors and consumers to express and share parsing logic in a standard format
  - Simplify product development
  - A way to change a native log into a standard format (example Apache to CEE)
  - Combine multiple log and data sources together into common output

Notional Architecture



Data Transformation

- OEEL would have three primary moving parts for performing the data mapping
  - A parser for parsing various input formats
  - A profile in the form of a markup or language that defines rules used to convert an input format to an output format
  - A transformer for actually transforming an input format to an output format based on a profile

Example (FFE – Flat File Extractor)

```
structure log {
  type separated " "
  quoted
  output cee
  record apache {
    field src-ip
    field src-host
    field acct-name
    # In CEE the time+timezone should be expressed at ISO8601 timestamp
    field event-time
    field event-timezone
    field http-request
    field http-status
    field trans-size
    field http-referrer
    field http-useragent
  }
}
output cee {
    # data "%D"
    indent "\t"
    file_header "<Log>\n"
    record_header "<Event>\n"
    data "<Field name=\"%n\">%d</Field>\n"
    record_trailer "</Event>\n"
    file_trailer "</Log>\n"
    # justify =
    # indent " "
}
```

Example (NOTIONAL)

```
<?xml version="1.0" encoding="UTF-8" "?>
<oeel:configuration xmlns:oeel="http://nist.g2-inc.com/oeel/">

<structure name="ApacheLog">
                <type name="seperated" value=",">
                <quoted name="true" value="">
                <output value="XML">
                <record name="apache">
                <param name="field" value="ipaddr" size="15">
                <param name="field" value="client" size="20">
                <param name="field" value="uid" size="10">
                <param name="field" value="date" size="25">
                <param name="field" value="client" size="20">
                <param name="field" value="timezone" size="10">
                <param name="field" value="request" size="512">
                <param name="field" value="status" size="10">
                <param name="field" value="size" size="10">
                <param name="field" value="referrer" size="512">
                <param name="field" value="userAgent" size="512">
                </record>
</structure>
                <output value="XML">
                <param name="file_header" value="<?xml version=\"1.0\" encoding=\"ISO-8859-1\"?>\n<%s>\n">
                <param name="data">
                <param name="record_header" value="<%r>\n">
                <param name="record_trailer" value="</%r>\n">
                <param name="indent" value="  ">
                <param name="file_trailer" value="</%s>\n">
   </output>
```

Flexibility

- For a specification to be effective it needs be flexible enough to express enough parsing logic to be useful
  - Feasibility still being studied
  - Many cases to be considered
  - A 100% solution here seems unattainable, but can we cover enough.
  - Need to identify MUST have cases and those that are less critical

Issues

- Some logs are just too messy to be considered here (at least at first).
  - If there is no discernable pattern or format
  - If it is a monumental programming task to parse a log, it probably isn't a good fit for a generic expression
  - BUT, there are plenty of logs that have a discernable format.
  - The most commonly occurring platforms and devices should be targeted first

Content Creation

- Who will do it?
  - Vendors
  - Community
  - Government
- Content creation will be a key issue
  - If no content exists, there will be no adoption
  - What incentivizes content production?

Content Reduction

- What about lossiness (lost in translation)?
  - How do we ensure content reduction does not occur?
  - Who is responsible for ensuring content reduction does not occur?
  - What should the interpreter do when encountering various errors
    - Wrong format
    - Un-parsed data

Content Protection

- What if I DON'T want to share?
  - Content is proprietary
  - Content is classified
  - Content exposes vulnerability
  - Should the specification allow for encrypted content (does this even help)?
  - Variables appear necessary in general, do they help here?
  - What other cases of "protecting" content can we envision?

Summary

- The number of log formats is staggering
- The number of parsers just as staggering
- We need a way to abstract parsing to share information
- Provides a method to normalize disparate log formats based on an open specification

Questions / Comments?

George Saylor
george.saylor@nist.gov