Timeline Of Learner Success

As learners engage in a blended eLearning ecosystem, they will build up a history of learning experiences. When that eLearning ecosystem adheres to a framework dedicated to supporting and understanding the learner, such as the Total Learning Architecture (TLA), it becomes possible to retell their story through data. One important aspect of that story is the learner's history of success.

1 Ideal Statements

In order to accurately portray a learner's timeline of success, there are a few requirements of the data produced by a Learning Record Provider (LRP). They are as follows:

- the learner must be uniquely and consistently identified across all LRPs
- learning activities which access a learner's understanding of material should report if the learner was successful or not
  - if the assessment is scored, the grade earned by the learner should be reported
  - if the assessment is scored, the minimum and maximum possible grade should be reported
- The learning activities must be uniquely and consistently identified across all LRPs
- The time at which a learner completed a learning activity must be recorded
  - The timestamp should contain an appropriate level of specificity.
  - ie. Year, Month, Day, Hour, Minute, Second, Timezone

1.1 statement parameters to utilize

The statement parameter locations here are written in JSONPath

- $.timestamp
- $.result.success
- $.actor
- $.verb.id

2 TLA Statement problems

The data collected at the TLA pilot run supports the following algorithm.

3 Algorithm

3.1 Summary

1. Query an LRS via a GET request to the statements endpoint using the parameters agent, since and until
2. Filter the results to the set of statements where:
   - $.verb.id is one of:
     - http://adlnet.gov/expapi/verbs/passed
     - https://w3id.org/xapi/dod-isd/verbs/answered
     - http://adlnet.gov/expapi/verbs/completed
   - $.result.success is true

3.2 Query an LRS via REST

How to query an LRS via a GET request to the Statements Resource [1]

```
Agent = "agent={"account":
                  {"homePage": "https://example.homepage",
                   "name": 123456}}"

Since = "since=2018-07-20T12:08:47Z"

Until = "until=2018-07-21T12:08:47Z"

Base = "https://example.endpoint/statements?"

endpoint = Base + Agent + "&" + Since + "&" + Until

Auth = Hash generated from basic auth

S = curl -X GET -H "Authorization: Auth"
         -H "Content-Type: application/json"
         -H "X-Experience-API-Version: 1.0.3"
         Endpoint
```

3.3 Z Specifications

3.3.1 xAPI Schema

[Statement] [Actor] [Verb] [Object] [Result] [Context] [Timestamp]

[1] S is the set of all statements parsed from the statements array within the HTTP response to the Curl request. It may be possible that multiple Curl requests are needed to retrieve all query results. If multiple requests are necessary, S is the result of concatenating the result of each request into a single set

Statement

$$s : Statement$$

$$\begin{aligned} s = \{&Actor, Verb, Object, Timestamp\} \mid \\ &\{Actor, Verb, Object, Timestamp, Context\} \mid \\ &\{Actor, Verb, Object, Timestamp, Result\} \mid \\ &\{Actor, Verb, Object, Timestamp, Result, Context\} \end{aligned}$$

- The variable s is of type Statement and consists of an Actor, Verb, Object, Timestamp and optionally Context and Result

Statements

$$S : Statements$$

$$S = \{\, s : Statement \mid S \neg \emptyset\}$$

- The variable S is of type Statements and is a set of objects s, each of type Statement
- The variable S is a non empty set

3.3.2 Timeline Leaner Success System State

TimelineLearnerSuccess

$$S_{extra}, S_{completion}, S_{success}, S_{failure} : \mathbb{P}\, S$$

$$S_{extra} \cup S_{completion} = S$$
$$S_{extra} \cap S_{completion} = \{\}$$
$$S_{success} \cup S_{failure} = S_{completion}$$
$$S_{success} \cap S_{failure} = \{\}$$

- The sets S˙extra, S˙completion, S˙success, S˙failure are the powerset of S
- The union of sets S˙extra and S˙completion is equal to the complete set of statements S
- No values are shared between the sets S˙extra and S˙completion
- The union of sets S˙success and S˙failure is equal to the set S˙completion
- No values are shared between the sets S˙success and S˙failure

3.3.3 Initial State of Timeline Learner Success System

$$\begin{array}{|l}
\multicolumn{1}{l}{\_\, InitTimelineLearnerSuccess \,\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_} \\
TimelineLearnerSuccess \\
\hline
S_{extra} = \{\} \\
S_{completion} = \{\} \\
S_{success} = \{\} \\
S_{failure} = \{\} \\
\hline
\end{array}$$

- The sets S˙extra, S˙completion, S˙success, S˙failure are all initially empty

3.3.4 Filter for Completion

$$\begin{array}{|l}
\multicolumn{1}{l}{\_\, VerbIdCompletion \,\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_} \\
V_{completion} : Verb \\
\hline
V_{completion} = http://adlnet.gov/expapi/verbs/passed \mid \\
\qquad\qquad\quad https://w3id.org/xapi/dod-isd/verbs/answered \mid \\
\qquad\qquad\quad http://adlnet.gov/expapi/verbs/completed \\
\hline
\end{array}$$

- The var V˙completion has a value of one of the above IRIs and is of type Verb

$$\begin{array}{|l}
\multicolumn{1}{l}{\_\, FilterForCompletion \,\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_} \\
\Delta TimelineLearnerSuccess \\
\hline
S'_{completion} = \{\, s : Statement \mid V_{completion} \in s \land s \in S\} \\
S'_{extra} = \{\, s : Statement \mid V_{completion} \notin s \land s \in S\} \\
\hline
\end{array}$$

- The updated set S’˙completion is the set of all statements s where V˙completion is in s and s is in S
- the updated set S’˙extra is the set of all statements s where V˙completion is not in s and s is in s

3.3.5 Filter for Success

$$\begin{array}{|l}
\multicolumn{1}{l}{\_\, ResultSuccessTrue \,\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_} \\
R_{successful} : Result \\
\hline
R_{successful} = true \\
R_{successful} \neq false \\
\hline
\end{array}$$

- The var R˙successful has a value of true but not false and is of the type Result

$$\begin{array}{|l}
\hline \mathit{FilterForSuccess} \\
\Delta \mathit{TimelineLearnerSuccess} \\
s_{completion} : \mathit{Statement} \\
\hline
s_{completion} \in S_{completion} \\
S'_{success} = \{ s_{completion} : \mathit{Statement} \mid R_{successful} \in s_{completion} \} \\
S'_{failure} = \{ s_{completion} : \mathit{Statement} \mid R \notin s_{completion} \} \\
\hline
\end{array}$$

- The set s˙completion is of type Statement and is in the set S˙completion
- The updated set S˙success' is the set of all statements s˙completion where R˙successful is in s˙completion
- The updated set S˙failure' is the set of all statements s˙completion where R˙successful is not in s˙completion

3.3.6 Return

$$\begin{array}{|l}
\hline \mathit{Return} \\
\Xi \mathit{TimelineLearnerSuccess} \\
S_{success}! : \mathit{Statements} \\
\hline
S_{success}! = S_{success} \\
\hline
\end{array}$$

- The returned variable S˙success! is equal to the current state of variable S˙success

3.4 Pseudocode

```
Input: S
Result: S-success
while S is not empty do
    for each Statement s in S
    if s.verb.id = V-completion then
        add s to S-completion
    else
        add s to S-extra
    end [2]
end
while S-completion is not empty do
    for each Statement s-completion in S-completion
    if s-completion.result.success = R-success then
        add s-completion to S-success
    else
        add s-completion to S-failure
    end
end
```

3.5 Result JSON Schema

3.6 Visualization Description

description of the associated visualization in english

3.7 VEGA example

This section will be updated to include a VEGA JSON blob for prototype viz

[2] S˙completion = S-completion s˙completion = s-completion, S˙extra = S-extra, V˙completion = V-completion, R˙successful = R-successful, S˙success = S-success, S˙failure = S-failure