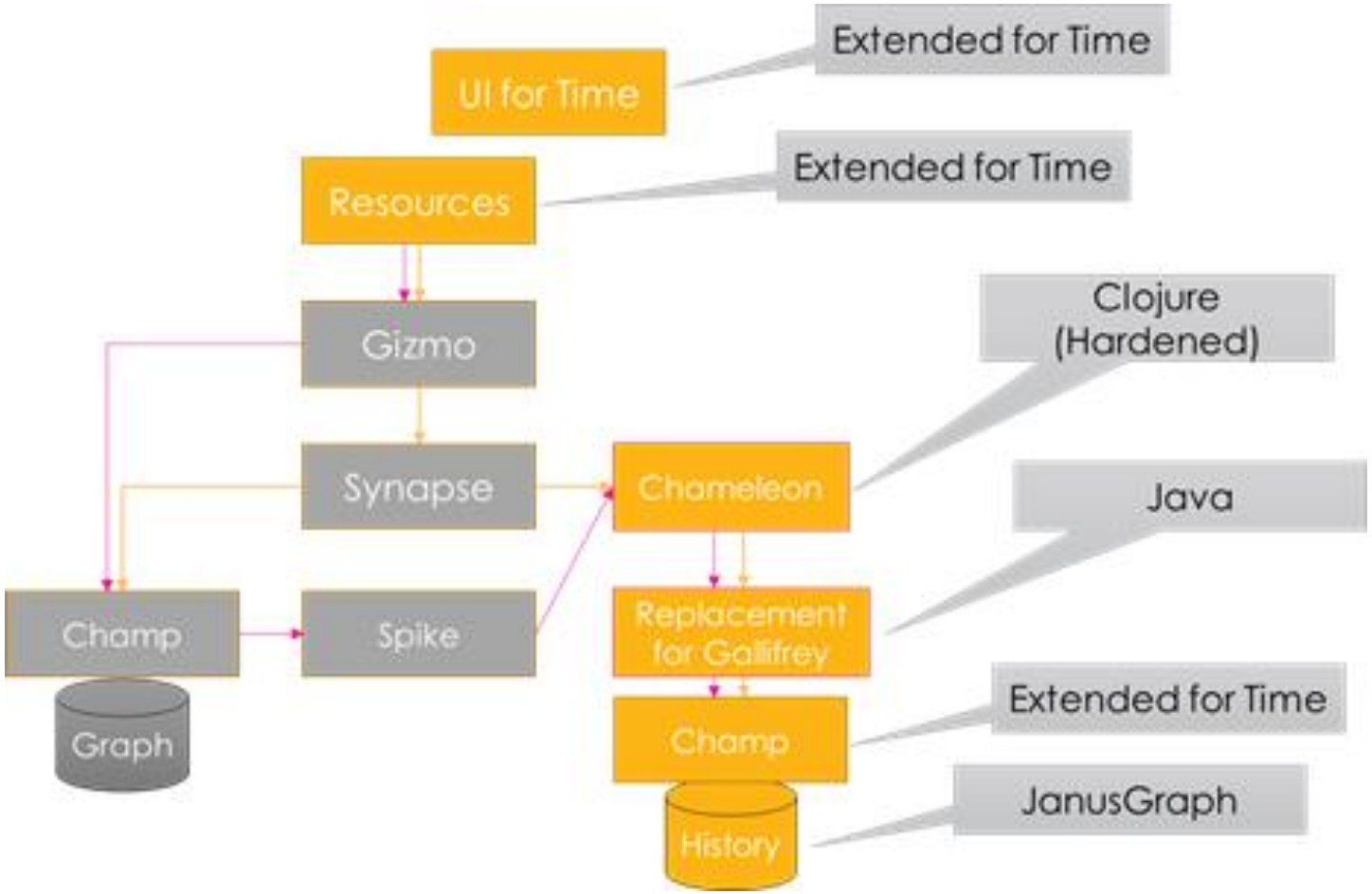
Historical Data

t-t vs t-k

Historical Data Microservice Flow



Recording Time Values

- There are two values of time that are of interest for each event:
  - 'Truth' (t-t) … when the client says it happened (network time)
  - 'Knowledge' (t-k) … when the system records it (database time)

- Question … when do we record t-k value? Two logical options to consider:
  - When writing to the 'live' database
  - When writing to the 'historical' database

Laws Of Historical Data

1. Want historical data stored in a separate database
2. Who ever is responding to the database/knowledge time question should be the one setting the database/knowledge time value
3. Database/knowledge time should never change

Scenario Setup

- Desire: want to evaluate when is the preferred place to record the t-k value
- Initial Setup: Assume element X has an attribute a=1 initially
- At t0, client application will change value of ‘a’ to be 4
- Operating in a distributed system with the realization that there are some delays between time client requests an update and when the final write to the database happens … here are some critical times to consider:
    - T_{+0} – time the client performs the event
    - T_{+1} – time when live database records the event
    - T_{+2} – time when historical databse records the event

Option 1 – Live Database Identifies t-k Value

- At t_0
  - Ask what value of 'a' is at t_0 … 1
- At t_1
  - Client receives notification that value of 'a' changed to 4
  - Ask what value of 'a' is at t_0 … 1
  - Ask what value of **'a' is at t_1 … 1**
- At t_2
  - Ask what value of 'a' is at t_0 … 1
  - Ask what value of **'a' is at t_1 … 4**
  - Ask what value of 'a' is at t_2 … 4

Option 2 – Historical Database Identifies t-k Value

- At t_0
  - What is value of 'a' at t_0 … 1
- At t_1
  - Client receives notification **at t_1 … value of 'a' changed to 4**
  - Ask what value of 'a' is at t_0 … 1
  - Ask what value of 'a' is at t_1 … 1
- At t_2
  - Ask what value of 'a' is at t_0 … 1
  - Ask what **value of 'a' is at t_1 … 1**
  - Ask what value of 'a' is at t_2 … 4

Pros & Cons

Both options present pros & cons … need to identify which is the lesser of two evils (which are we more comfortable with)

	Pros	Cons
Live Database Record t-k	Use timestamp of response acknowledgement (200 code signifying live database wrote value), will always return the matching value of 4 going forward	Historical value will change over time
Historical Database Record t-k	Historical value will be consistent over time	Using timestamp of the response does not take into consideration propagation delays and thus will return the previous value

Discussion Points…

What are we optimizing the Historical Data feature for?

- Availability – timestamp from the 200 acknowledgement around writing to live database reflect accurate value in historical database
- Consistency – historical database shall always return a consistent value for a given timestamp

Truth vs Knowledge Example

- On Monday I purchased a dog and didn't tell you
- On Wednesday I sent you an email telling you I purchased a dog on Monday
- If I asked you on Tuesday if I had a dog, what would be your answer?
    - Truth is I had one, but to your knowledge I didn't
- What would your answer be if I asked you on Wednesday if I had a dog on Tuesday?
    - Knowledge is immutable in relation to time
    - According to truth, you answer I had one
    - According to knowledge, you didn't know I had a dog on Tuesday, and your answer about what you knew Tuesday should remain consistent

Alternate Design Considerations

- Wrap writing to both databases in a single transaction and only 200 acknowledge once the transaction has completed
- Spike events for writing to live database does not propagate 200 acknowledgement all the way back to client … add new spike after write to historical database and propagate 200 acknowledgement to the client then

Both solutions above keep historical database data consistent over time and ensures the timestamp on the 200 response code lines up with any query to the historical database … but it comes at the cost of complexity and/or changes to existing architecture.