PDP-6: State & health in Application Management

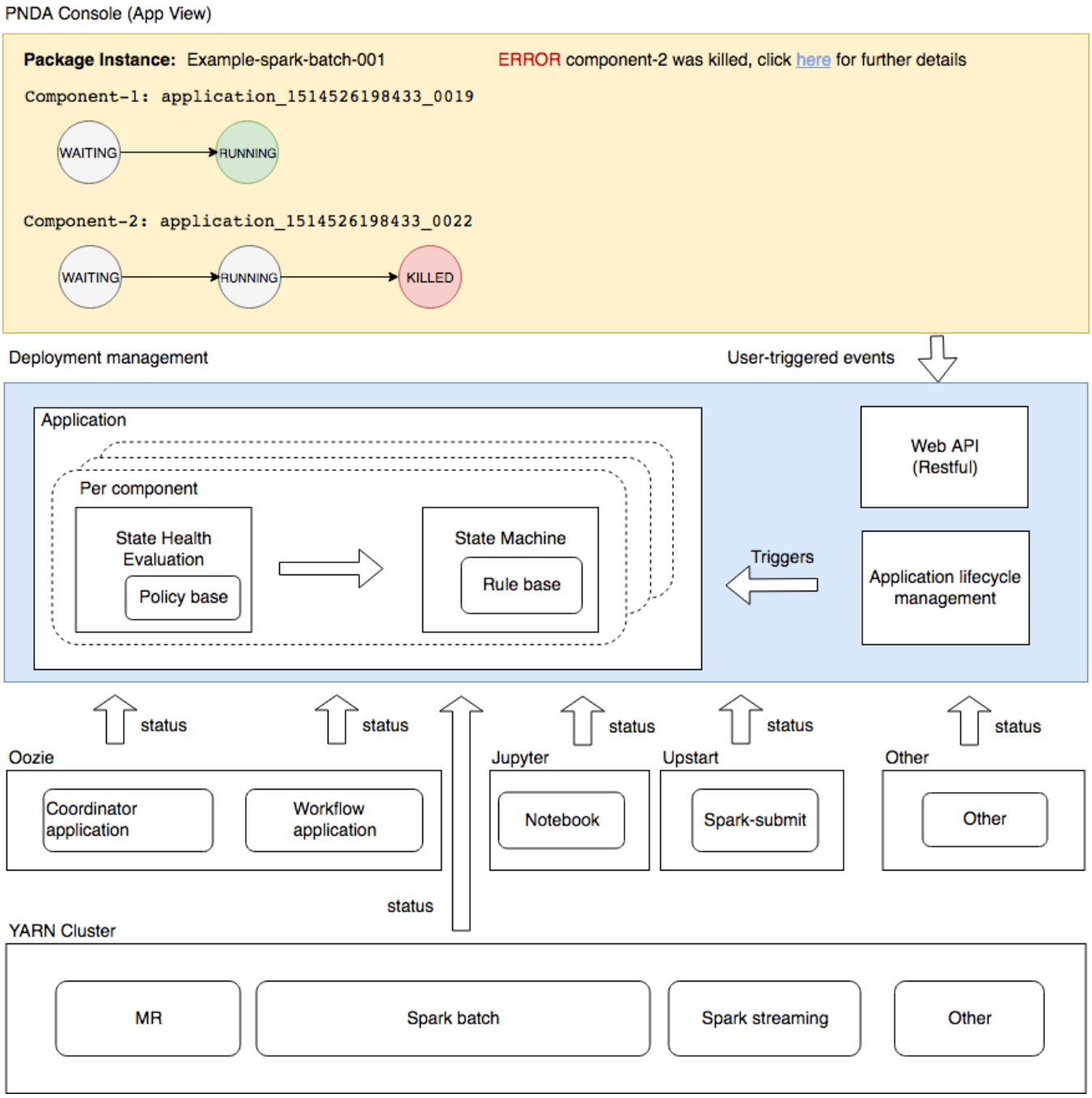
Motivation

- We have a lot of open bugs related to application status reporting that can be traced to a common set of problems
    - Current design does not have a good separation of key concepts by conveniently choosing state reporting from underlying technologies (e.g. Oozie, YARN, etc.).
    - A lack of consensus on state definitions/semantics at high level has resulted in ad-hoc state reporting on different application types, making its implementation unmaintainable and inextensible.
        - Net result is that it's often not clear to users exactly what state an application is in, and the same state sometimes means different things depending on the type of application
    - The link from an application to the explanation of why an application is that state - or what an operator should do about it - is not sufficiently intuitive

Proposal

Essentially, PNDA needs a better high-level abstraction of application states and transitions on per composite component basis. This needs to include -

- Analysis of all PNDA component types and their states and transitions provided by their underlying technologies
- Design of an abstraction that sits across the various different underlying component types
- Design of a rule base that is able to give sensible advice to the user based on the abstracted state and underlying metrics for the given application type
- Design of a better user interface that makes use of this abstraction and rule base to present both current state and 'call to action' links when things go wrong



State machine

- Abstract state machine that is simply enough to cover existed and futher underlying technologies.
- Detailed definitons of states, initial state, and state tranistions (source state, destination state, triggers, and conditions).

Health evaluation

- Define common health status.
- Add common features/metrics that applies to each state
- Core policies to evaluate states' health against reportable metrics from underlying technologies (e.g. asserting a long-running componet is running ok without failed stages).
- User-defined policies to evaluate state health against state-specific metrics (e.g. MaxPrepTimeAllowed=300 can be used to limit the maximum delay to 5 minutes before a component start running).

Deployment manager

- Major updates will be need to re-implement application lifecycle and status reporting in deployment manager.
- Application summary registrar will be deprecated.

Interface

- DM interface to retrieve application summary to be deprecated
- New interfaces to be documented for retrieving component state and health
- Console frontend to be updated for wrapped component state reporting/monitoring

Plan

- Stage 1
  - Agree on paper what managed entities and relations of underlying technologies and what are abstract states and state transitions. see 6.1 State abstraction for more details
  - Agree on paper what are key health status and basic evaluation rules on state health assertions.
- Stage 2 (TBD)
  - Implementation of state machine
  - Implementation of health evaluation (YARN)
- State 3 (TBD)
  - Update frontend console.