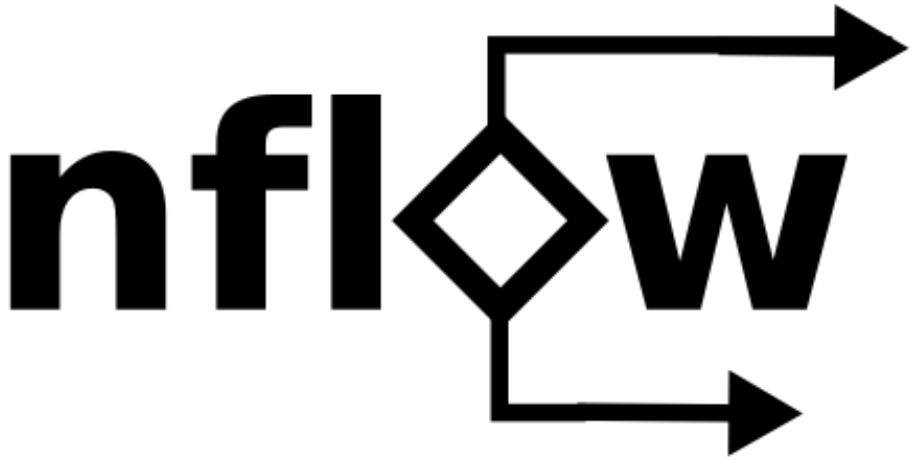


THE PRAGMATIC WORKFLOW FRAMEWORK

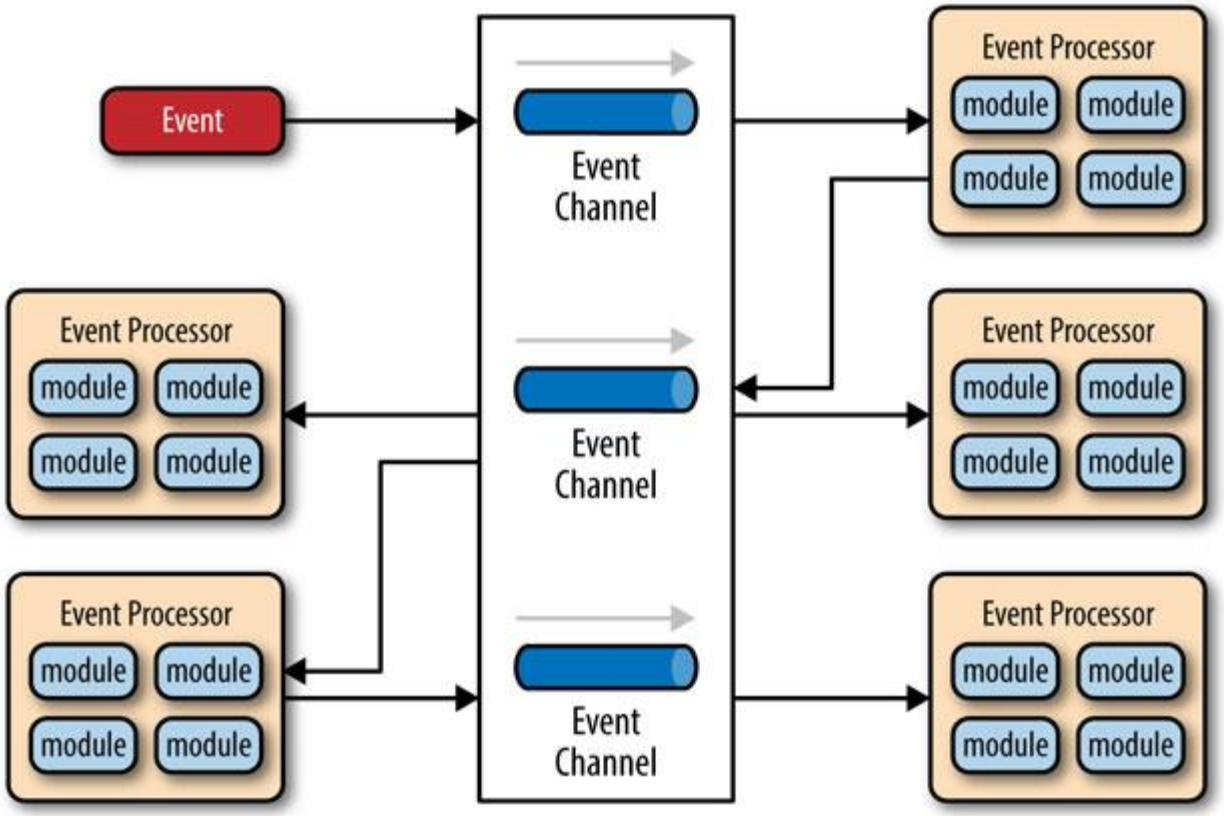
Agenda

1. Architectural Prologue
2. Gathering Motivation
3. Introducing nFlow
4. Example: Credit Application Workflow
5. Success Stories
6. Performance
7. Next Steps

Architectural Prologue – Event-driven Architecture

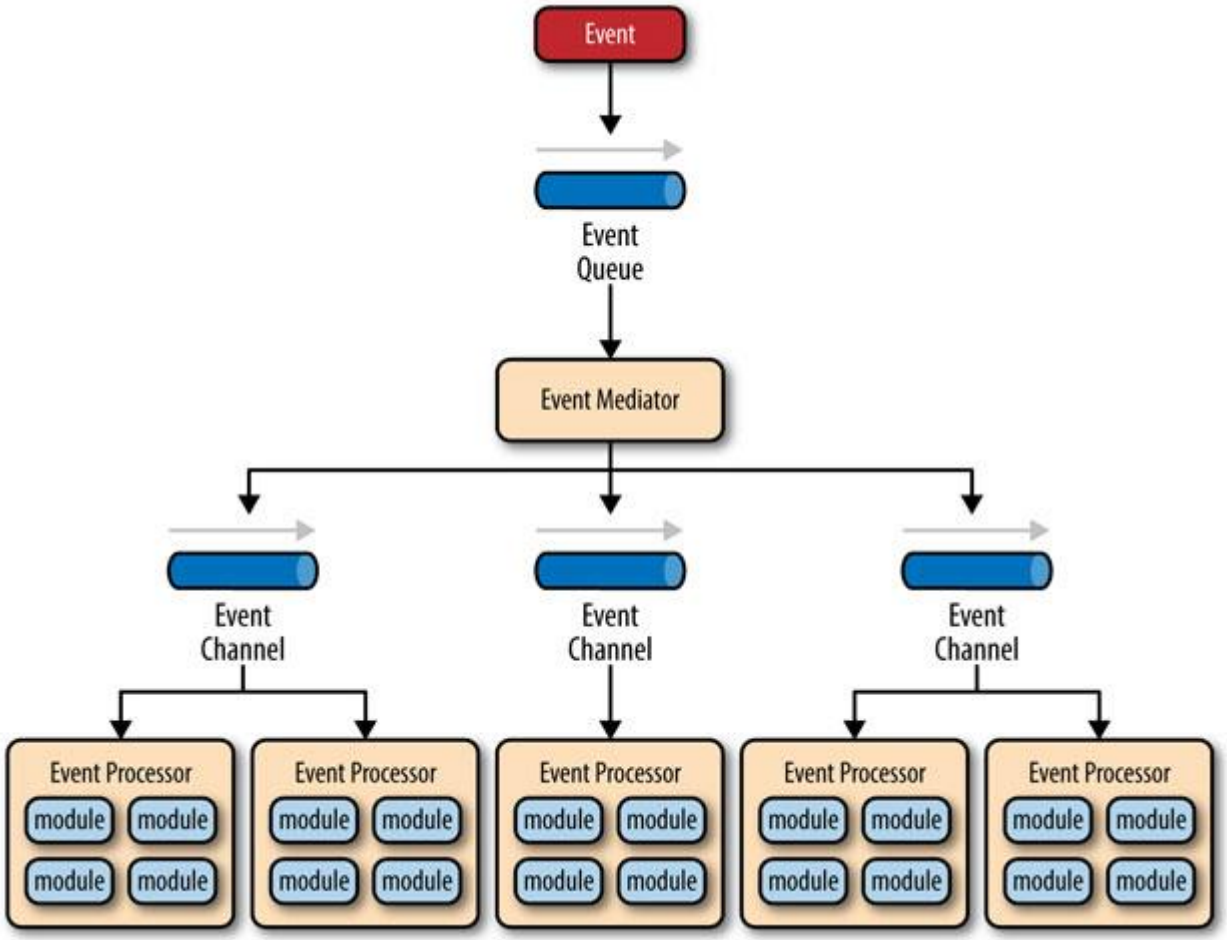
- “Event-driven architecture (EDA) is a software architecture pattern promoting the production, detection, consumption of, and reaction to events.” – Wikipedia
- Two main topologies
  - **Broker topology:** no central event mediator, workflow is distributed across the event processors as chain through message broker (ActiveMQ, HornetQ,…)
  - **Mediator topology:** has central event mediator that orchestrates the workflow (jBPM, Activiti, Mule ESB,…)
- nFlow facilitates event-driven architecture and implements hybrid of broker and mediator topologies
  - Business workflow follows broker topology
  - Technical errors and retrying handled by the framework

Architectural Prologue – Broker Topology



http://radar.oreilly.com/2015/02/variations-in-event-driven-architecture.html

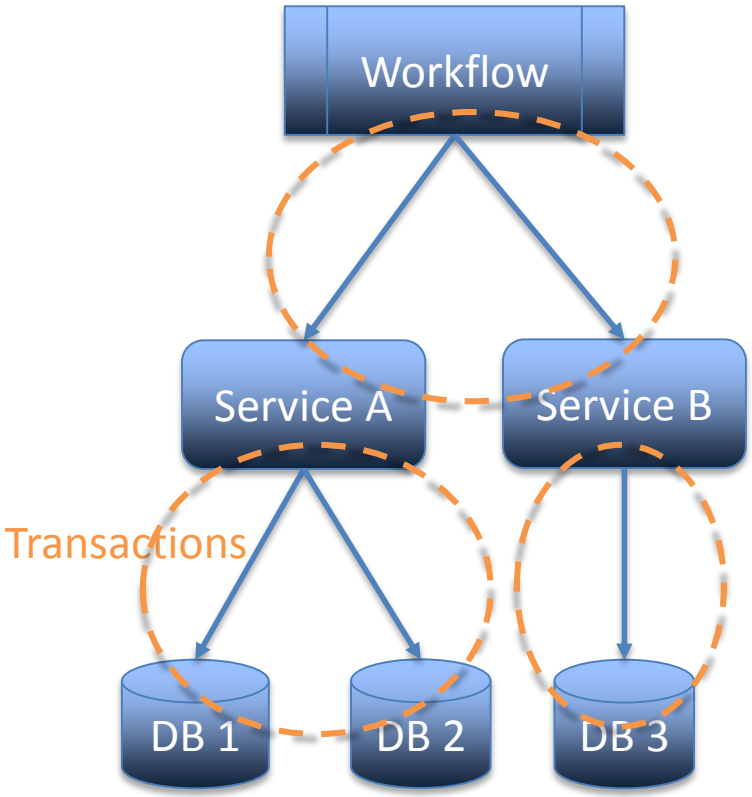
Architectural Prologue – Mediator Topology



http://radar.oreilly.com/2015/02/variations-in-event-driven-architecture.html

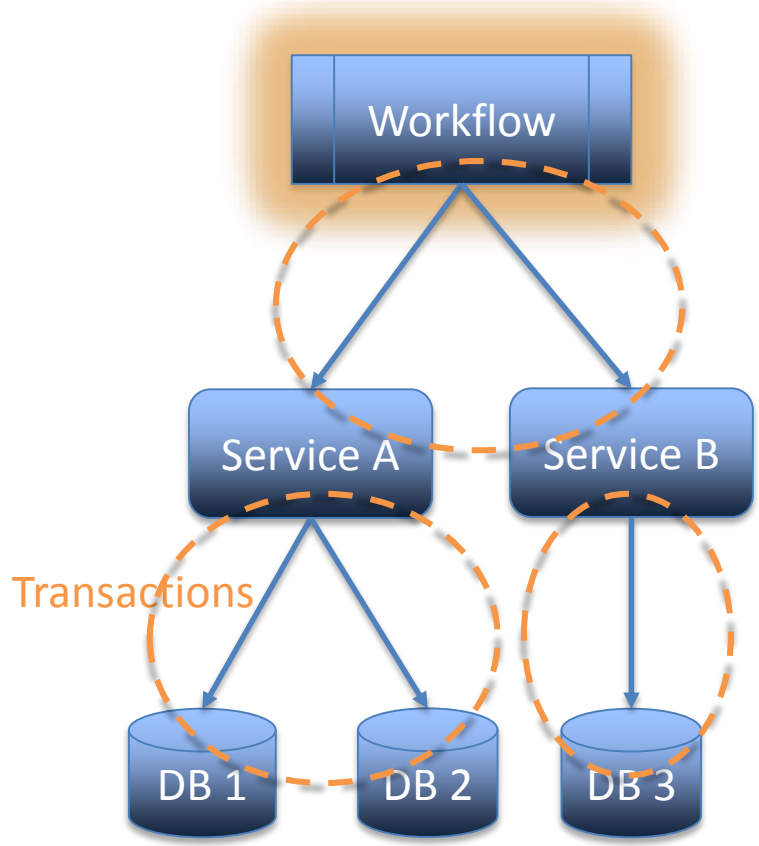
Gathering Motivation – Information Systems

- Information systems oftentimes contain following elements:
  - Workflows
  - Services
  - (Distributed) transactions
  - Databases
- Example:
  - **Deliver order** workflow uses **send payment** and **dispatch items** services
  - Payment and delivery statuses are stored to respective databases
- How do we implement them?



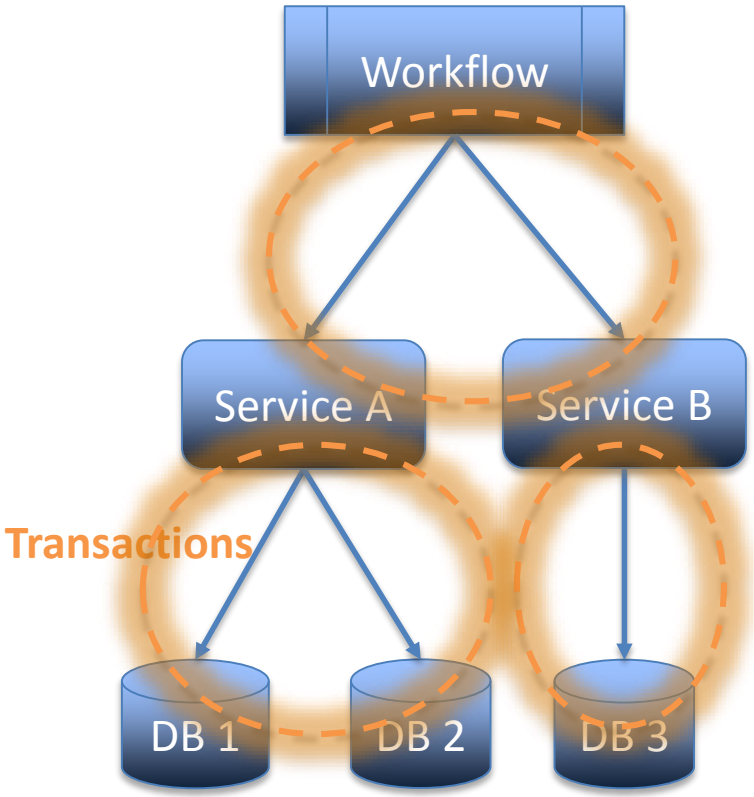
Gathering Motivation – Workflows

- Workflows are usually implicit and hardcoded in the user interface and/or services layer
  - An example: user action starts a synchronous chain of service calls, whose result determines the next workflow state
  - Outcome: bad user experience and obfuscated business process (=read whole codebase in order to understand the workflow)
- Other ways to implement workflows:
  - Collection of queues
  - Process engines (anyone?)
  - Custom workflow engines



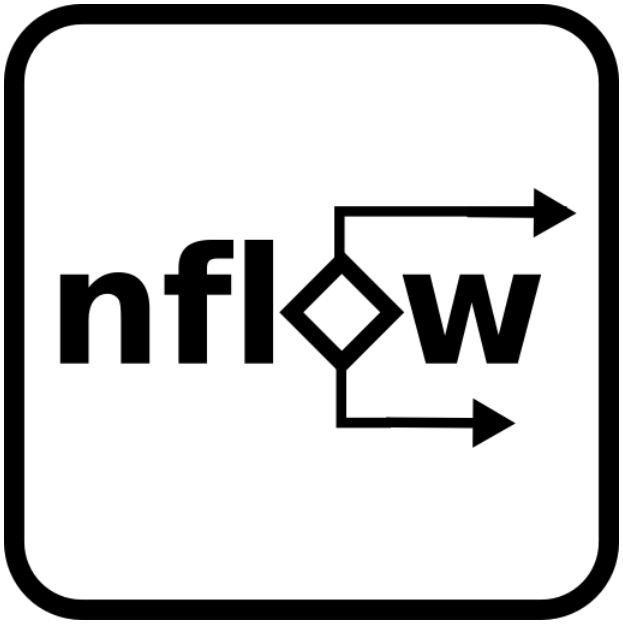
Gathering Motivation – Transactions

- Transactions guarantee that ACID properties are fulfilled. This is all good and well, but…
- Distributed transactions that span over multiple resources (XA) are painful to manage and exact a performance penalty
- Many integration technologies (e.g. trending REST services) do not enable distributed transactions that span over multiple services



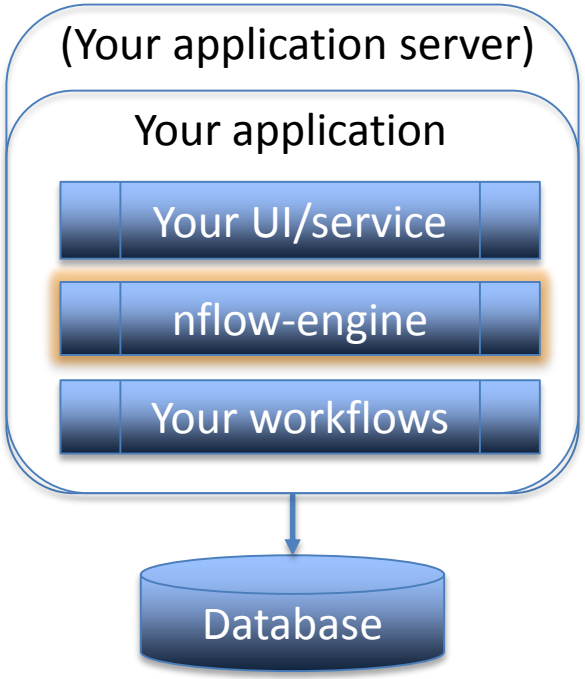
Introducing nFlow

- nFlow is a lightweight and modular solution for orchestrating workflows using Java
  - Explicit workflow implementation as components
  - Multiple management and monitoring options (UI, REST, …)
  - Low entry barrier: single Java-library + few database tables
- Promotes micro-service architectures in which:
  - Reliable outcomes are guaranteed by idempotent retry pattern (instead of distributed transactions)
  - Good user experience is achieved through request/acknowledge pattern



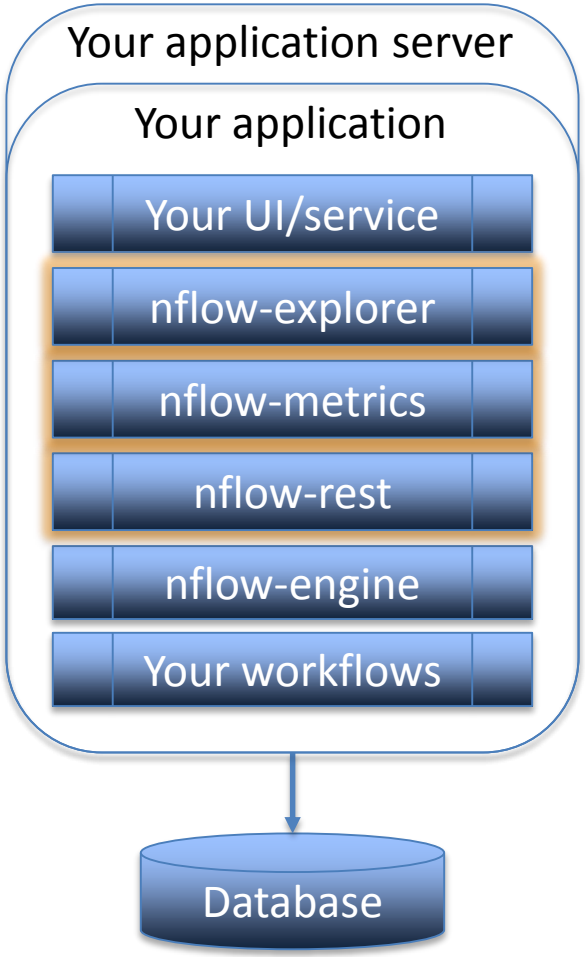
Introducing nFlow – Minimal Setup

- In a minimal setup:
  - Embed a single library to your application (nflow-engine)
  - Create few database tables
    - Supported databases: PostgreSQL, MySQL, MariaDB, H2
  - Implement your workflow component(s)
  - That's it!



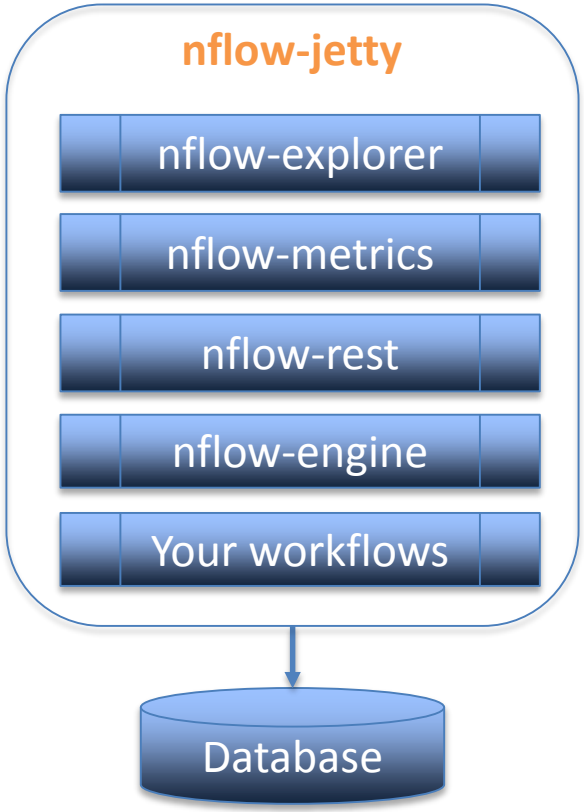
Introducing nFlow – Optional Modules

- Optional modules include
  - **nflow-explorer**
    - Management and monitoring UI
  - **nflow-rest**
    - JAX-RS compliant REST API for workflow management and monitoring
    - Swagger UI for API testing and documentation
  - **nflow-metrics**
    - Integration to monitoring tools like Graphite and Ganglia



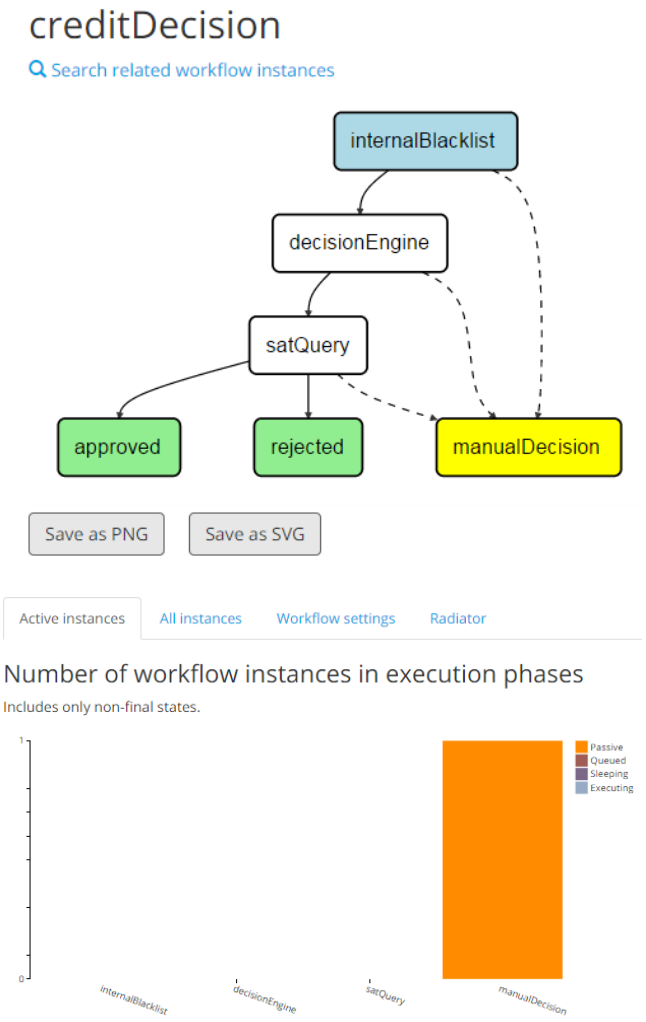
Introducing nFlow – Standalone Server

- Standalone server (nflow-jetty) is also provided, if you want keep your workflows separated from existing applications
- Useful for quick-start evaluation of nFlow



Introducing nFlow – Explorer

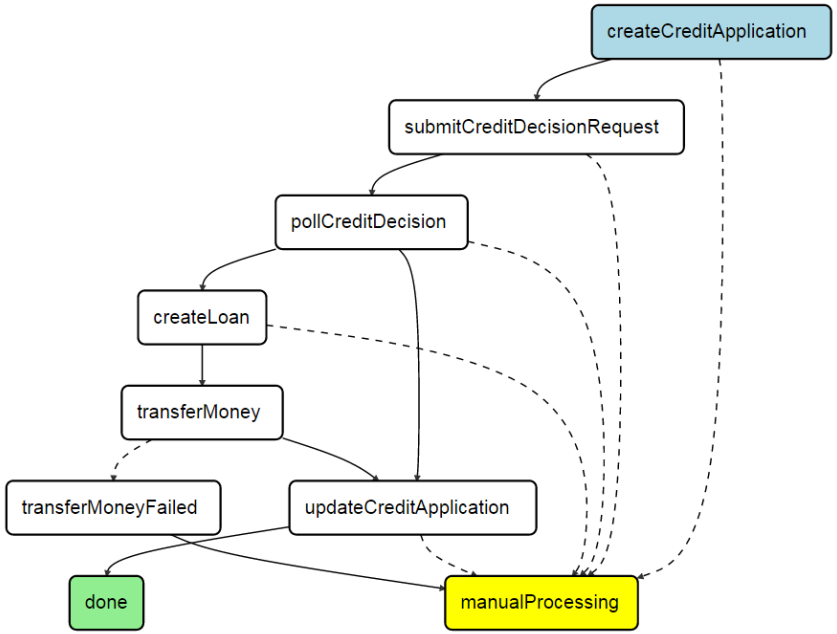
- nFlow Explorer is a monitoring and management UI
  - Search, manage and visualize workflow instances
  - Visualize workflow definitions
  - Visualize and monitor instance statistics per workflow definition
- JavaScript application that uses nFlow REST services



nitorcreations

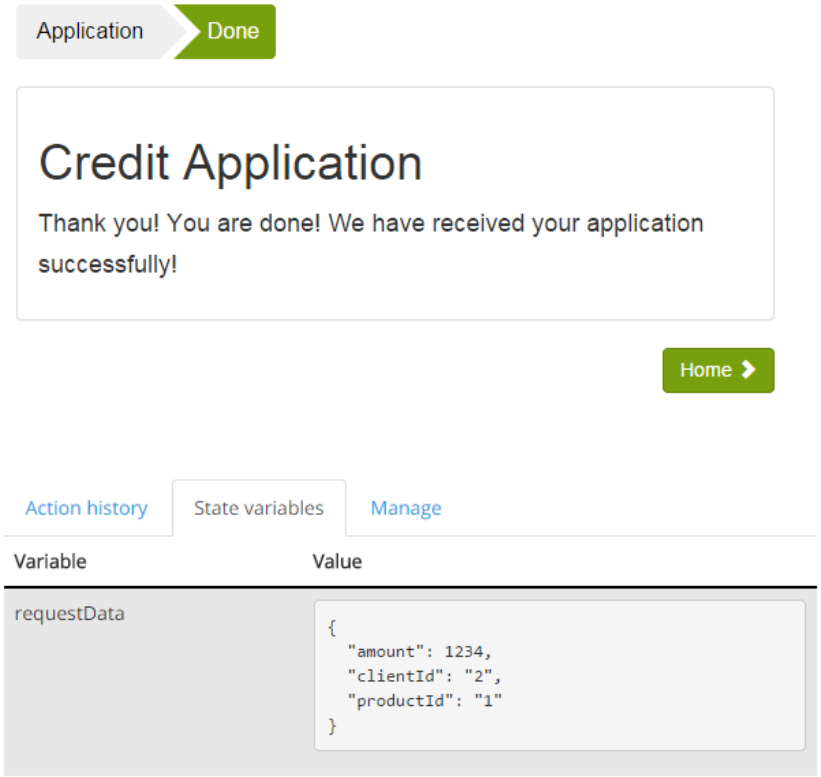
Example: Credit Application Workflow (1/3)

- The workflow orchestrates processing of a single credit application
  1. New workflow instance is created, when a customer submits an application
  2. The result of credit decision (sub)workflow determines if money is transferred or application rejected
  3. Each step is retried until successful, persistent failures lead to manual processing



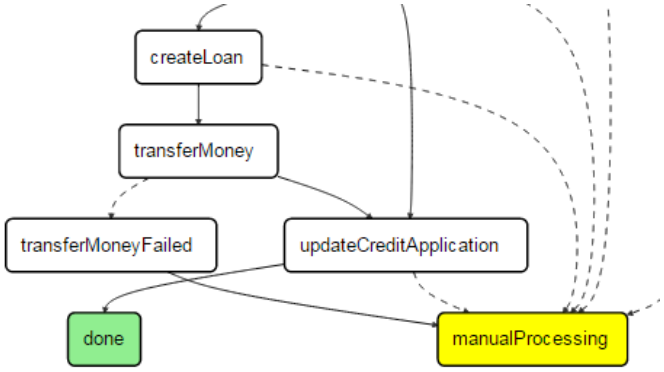
Example: Credit Application Workflow (2/3)

- Application of request/acknowledge pattern
  - Workflow instance is created with the credit application information
  - All backend system updates are done via the workflow
  - The only UI dependencies are cached business data (read only) and nFlow



Example: Credit Application Workflow (3/3)

- Application of idempotent retry pattern
  - Variation of classical money transfer example
  - Idempotent retry takes care of realiable outcome instead of transaction
  - Manual work is required, if the operation fails consistently over long period of time
  - In transactional solution support organization would be contacted immediately



Action history | State variables | Manage

No	State	Description	Retries	Started	Finished	Duration
8	updateCreditApplication	Credit application updated	0	Jan 27, 2015 2:34:01 PM	Jan 27, 2015 2:34:01 PM	10 msec
7	transferMoney	Money transferred	1	Jan 27, 2015 2:34:00 PM	Jan 27, 2015 2:34:01 PM	6 msec
6	transferMoney	com.sun.jersey.api.client.UniformInterfaceException: Client response status: 503	0	Jan 27, 2015 2:32:00 PM	Jan 27, 2015 2:32:00 PM	10 msec
5	createLoan	Loan created	0	Jan 27, 2015 2:32:00 PM	Jan 27, 2015 2:32:00 PM	5 msec
4	pollCreditDecision	Credit decision approved	0	Jan 27, 2015 2:32:00 PM	Jan 27, 2015 2:32:00 PM	8 msec

Success Stories

- Order management and delivery processes for a large multinational media company
  - Example workflow: process new order
    - Workflow instance is created, when a customer submits her order. After the payment confirmation, the workflow fulfills all order entries, updates other backend system (e.g. customer master) and notifies external systems.
    - Workflow definition contains about 20 steps
  - Millions of processed workflow instances per year, peak rate over 20 started workflows per second

Performance

- In a typical setup, nFlow can process 10000-50000 workflow instances per minute
  - Typical setup means one database server and two application servers running nFlow that process the same workflow instances
  - The actual throughput is determined by the complexity of the workflow steps
- nFlow application servers can be scaled horizontally up to a point where the database becomes the bottleneck. After this you have two options:
  - Get a faster database (e.g. more IOPS from your cloud provider or faster SSD drives)
  - Distribute your workflows to multiple executor groups that use different databases
- nFlow performance data for AWS is available

Next Steps

- Test drive nFlow by following the 1 minute guide
- Read the documentation
- Start using nFlow in your own projects
- Missing a feature? Contact us through Google group (nFlow-users)!