OpenWhiskProposal

OpenWhisk Proposal

OpenWhisk is an open source, distributed Serverless computing platform able to execute application logic (Actions) in response to events (Triggers) from external sources (Feeds) or HTTP requests governed by conditional logic (Rules). It provides a programming environment supported by a REST API-based Command Line Interface (CLI) along with tooling to support packaging and catalog services.

Champion: Sam Ruby, IBM

Mentors:

- Felix Meschberger, Adobe
- Isabel Drost-Fromm, Elasticsearch GmbH
- Sergio Fernández, Redlink GmbH

Background

Serverless computing is the evolutionary next stage in Cloud computing carrying further the abstraction offered to software developers using Container-based operating system virtualization. The Serverless paradigm enables programmers to just “write” functional code and not worry about having to configure any aspect of a server needed for execution. Such Serverless functions are single purpose and stateless that respond to event-driven data sources and can be scaled on-demand.

The OpenWhisk project offers a truly open, highly scalable, performant distributed Serverless platform leveraging other open technologies along with a robust programming model, catalog of service and event provider integrations and developer tooling. Specifically, every architectural component service of the OpenWhisk platform (e.g., Controller, Invokers, Messaging, Router, Catalog, API Gateway, etc.) all is designed to be run and scaled as a Docker container. In addition, OpenWhisk uniquely leverages aspects of Docker engine to manage, load balance and scale supported OpenWhisk runtime environments (e.g., JavaScript, Python, Swift, Java, etc.), that run Serverless functional code within Invoker compute instances, using Docker containers.

OpenWhisk's containerized design tenants not only allows it to be hosted in various IaaS, PaaS Clouds platforms that support Docker containers, but also achieves the high expectation of the Serverless computing experience by masking all aspects of traditional resource specification and configuration from the end user simplifying and accelerating Cloud application development. In order to enable HTTP requests as a source of events, and thus the creation of Serverless microservices that expose REST APIs, OpenWhisk includes an API Gateway that performs tasks like security, request routing, throttling, and logging.

Rationale

Serverless computing is in the very early stages of the technology adoption curve and has great promise in enabling new paradigms in event-driven application development, but current implementation efforts are fractured as most are tied to specific Cloud platforms and services. Having an open implementation of a Serverless platform, such as OpenWhisk, available and governed by an open community like Apache could accelerate growth of this technology, as well as encourage dialog and interoperability.

Having the ASF accept and incubate OpenWhisk would provide a clear signal to developers interested in Serverless and its future that they are welcome to participate and contribute in its development, growth and governance.

In addition, there are numerous projects already at the ASF that would provide a natural fit to the API-centric, event-driven programming model that OpenWhisk sees as integral to a Serverless future. In fact, any project that includes a service that can produce or consume actionable events could become an integration point with OpenWhisk-enabled functions. Apache projects that manage programming languages and (micro) service runtimes could become part of the OpenWhisk set of supported runtime environments for functions. Device and API gateways would provide natural event sources that could utilize OpenWhisk functions to process, store and analyze vast amounts of information immediately unlocking the potential of fast-growing computing fields offered in spaces as IoT, analytics, cognitive, mobile and more.

Initial Goals

OpenWhisk is an open source community project which seeks to adopt the Apache way through the course of the incubator process and foster collaborative development in the Serverless space.

Currently, the OpenWhisk project's source repository is in GitHub using its associated project tooling, but we believe the open Apache processes, democratic project governance, along with its rich developer community and natural integrations with existing projects provide the ideal fit for the technology to grow.

Serverless will only reach its full potential and avoid fragmentation if it is grown in an environment that Apache can offer.

Current Status

The OpenWhisk project was published as an open source project within GitHub (https://github.com/openwhisk) under the Apache v2.0 license in February 2016. The project consists of the “core” platform repository (https://github.com/openwhisk/openwhisk) code along with its family of repositories that include a “catalog” of OpenWhisk system and utility packages.

The project also includes repositories for:

- JavaScript and Swift SDKs for client integration

- Docker SDK for user-created “blackbox” (Action) runtimes
- Graphical Command Line Tutorial (using NodeJS)
- Packages for popular service integrations (i.e., JIRA, Twilio, Slack, Kafka, RSS, etc.)

Issue tracking and project governance (milestones, epics) are also managed through GitHub Issues and visualized through ZenHub. All “pull” requests, once passing automated tests run by TravisCI, are reviewed by “core” contributors with “write” privileges. IBM has also setup private staging servers to “stress” test the platform performance under load and over extended periods of time before being merged into the main code branch. As part of the incubation process we would make these staging tests public and have them be run by Apache.

Currently, the project is not officially versioned and is considered an “experimental beta”, but is marching towards milestone 10 that aligns with what is considered to be a “beta” the end of October and another milestone 11 end of November 2016 which is considered “GA” content for the “core” platform. Again, we would very much like to adopt an Apache community system for deciding on milestones, constituent epics (features) along with dates a versioning plan and communicate effectively using email lists, IRC and a project homepage (which is currently lacking).

In addition to the OpenWhisk core runtime, IBM and Adobe plan to collaborate and contribute to the API Gateway component under an open framework with the Apache community. The API Gateway Framework component would provide essential support for a Serverless environment including container services, platform services and traditional runtimes and provides functionality for API security, request validation, request routing, rate limiting, logging, caching and load balancing.

Meritocracy

The OpenWhisk project firmly believes in meritocracy from its inception. Issue, Feature and code submissions, to fix, improve or optimize the platform code, tooling and documentation, as well as contributions of new SDKs, Packages, Tutorials, etc. have all been welcomed after successful community input, consultation and testing. Contributions can be made by anyone as long as integration and staging (including stress and performance) tests pass. We are looking forward to talented individuals to progress the success of OpenWhisk and an open Serverless ecosystem surrounding it. It would be a pleasure to invite strong contributors to become committers in the project areas where they have shown a consistent track record.

Community

OpenWhisk has made significant effort to build a community using all possible media and social outlets as possible, always asking for interested developers to join and contribute.

The following outlets have been created to engage the public in as many ways as we could conceive. Every single of these sources is monitored continually via OpenWhisk code that triggers events and messages to appropriate developer Slack channels where we seek to respond and engage as quickly as we can.

- Twitter: https://twitter.com/openwhisk
- Slack: https://dwopen.slack.com/messages/openwhisk/
- StackOverflow: http://stackoverflow.com/search?q=OpenWhisk
- dwAnswers (developerWorks): https://developer.ibm.com/answers/smartspace/open/
- Blog site: https://developer.ibm.com/openwhisk/blogs/
- Google group: https://groups.google.com/forum/ - !forum/openwhisk

IBM has sought to promote OpenWhisk at every logical event worldwide where we are able.

Events and Meetups:
20+ past events, 6 planned through YE 2016 (across 12 countries)
Event calendar: https://developer.ibm.com/openwhisk/events/
Stats (GitHub):
43+ contributors: https://github.com/orgs/openwhisk/people
Contribution Graphs: https://github.com/openwhisk/openwhisk/graphs/contributors
Stars:
623 (and growing ~10-20 per week on average): https://github.com/openwhisk/openwhisk/stargazers

Core Developers

The following core developers, along with their credentials, are proposed; each have been committers within OpenWhisk since its initial development:

- Stephen Fink, sjfink@us.ibm.com, original project architect
- Rodric Rabbah, rabbah@us.ibm.com, project's developer who has deepest knowledge who has been with the project since its inception.
- Markus Thommes, markus.thoemmes@de.ibm.com, project build and deployment expert for all roles and environments (Mac, Linux, etc. either local/distributed).
- Jeremias Werner, JEREWERN@de.ibm.com, tooling and integration expert. Understands all the build and runtime dependencies / external projects OpenWhisk relies upon.
- Perry Cheng, perry@us.ibm.com, Performance and stress testing guru.

Alignment

We have looked, from the earliest days of developing OpenWhisk, at Apache as a model for building a strong developer community and worked to adopt its spirit and its best practices. From the outset, we have wished to have enough interest and momentum in order to have a robust pool of developers in order to adopt an Apache governance model for meritorious acknowledgement of committer and core contributors who can bring external knowledge to further grow the project.

We see immediate chances to leverage Apache projects such as Kafka, Camel, MQTT, ApacheMQ, etc. Wherever there is a collector, funnel or router of message data that can directly or indirectly generate events, we intend to link to OpenWhisk as an even provider. These and other projects are listed below and are just, we hope, “scratching the surface” of integration points for Serverless enabled applications.

In addition, we should note that we see immediate interest in leveraging the Apache relationship with the Linux foundation to integrate with the OpenAPI specification (f.k.a., Swagger) and seek to standardize API gateways that follow that spec. to formalize endpoints for services that can produce events.

Known Risks

Orphaned products

OpenWhisk and its initial group of committers along with the community currently supporting the project will continue to promote and look for ways to engage new developers and provide linkage to other compatible open source projects. Serverless computing has a significant future in Cloud computing and an open source implementation of a platform, as OpenWhisk embodies, must success to provide competition and interoperability and provide a rich foundation for new Serverless technologies to rely upon.

Inexperience with Open Source

OpenWhisk, as you can deduce from its name, has been an open source project from its public debut in February 2016. As soon as a the initial code, developed within IBM research, was viable and provided the functionality expected of a Serverless platform, the project team open sourced it and sought to build an open community to evolve it. Most all current all current project team members have strong experience developing within open source projects with meritorious governance models. In fact, several of the current team members are committers on other Apache projects and are excited to reach out to and align with other project communities within Apache.

Homogenous Developers

The current list of committers includes developers from two different companies. The current set of committers are geographically distributed across the U. S., Europe and China. All committers are experienced with working in a distributed environment and utilize many messaging and collaboration tools to continually communicate with each effectively to develop and review code regardless of location.

Additionally, the current project members are very focused on addressing comments, feedback and issue or feature requests as soon as we are able. In fact, we utilize OpenWhisk itself to intelligently notify project developers with the correct knowledge or expertise of any public posting to any community outlets (listed above).

Reliance on Salaried Developers

All of the initial developers are currently salaried by either IBM or Adobe. With increasing awareness and interest in Serverless technologies, we expect this to change due to the addition of volunteer contributors. We intend to promote and encourage participation whenever interest is shown in the project to build a robust community.

Relationships with Other Apache Products

Some possible project intersections or potential connections are listed below. We hope to identify many others through the course of incubation.

- Kafka, http://kafka.apache.org/project, OpenWhisk has plans to use Kafka for an intelligent “message hub” service that can channel events to OpenWhisk triggers.
- Camel, http://camel.apache.org/message-bus.html, Any message bus naturally carries message data that may carry events directly or be used indirectly to derive events that developers can link to OpenWhisk actions.
- ActiveMQ, http://activemq.apache.org/, Again, a widely used message server, that supports MQTT and AMQP, which can provide trusted event data to OpenWhisk.

Some additional projects we would like to explore any connection with include:

- CouchDB, https://projects.apache.org/project.html?couchdb: OpenWhisk already supports use of CouchDB for its own storage needs (Actions, Bindings, etc.); however, there may be more integrations possible as we develop a package manifest to describe OpenWhisk entities reposited in document stores as pseudo-catalogs.
- Mesos, https://projects.apache.org/project.html?mesos: in effect, OpenWhisk also manages a “pool of nodes” that can run various Actions (functions). It would be interesting to see if any overlap or sharing of node resources could be achieved.
- Spark, https://projects.apache.org/project.html?spark : As with Mesos, OpenWhisk nodes could be leveraged to perform distributed data-processing with Spark.

and many others that we hope the community will help identify and prioritize for development work.

An Excessive Fascination with the Apache Brand

The developers of OpenWhisk share a high appreciation of the Apache Software Foundation, and many have been active as users, contributors or committers to other Apache projects.

The main expectation for the developers is not the Apache brand, but the project governance and best practices established by the ASF, access to the Apache community and support and mentorship through senior Apache members.

Documentation

OpenWhisk offers a comprehensive set of documentation (primarily in Markdown) for all parts of the project from installation and deployment (locally, remotely, distributed) on various platforms in order to get developers “up and running” as quickly as possible on multiple platforms (Mac, Windows, Ubuntu). In addition, OpenWhisk goes to great links to document its architecture and programming model and provide guided tutorials for the CLI. All SDKs and Packages that can be installed, besides installation and use cases descriptions, often include videos and blogs. OpenWhisk is dedicated to providing the best documentation possible and even has volunteers’ submissions for translations in some areas.

Initial Source

The project is comprised of multiple repositories all under the primary openwhisk name. All initial source that would be moved under Apache control can be found in GitHub (by repository) here:

- Primary Repositories:
  https://github.com/openwhisk/openwhisk
  primary source code repository including run books, tests.
  https://github.com/openwhisk/openwhisk-catalog
  Catalog of built-in system, utility, test and sample Actions, Feeds and provider integration services and catalog packaging tooling.
- Client (SDK) repos.:
  https://github.com/openwhisk/openwhisk-client-js
  JavaScript (JS) client library for the OpenWhisk platform.
  https://github.com/openwhisk/openwhisk-client-swift
  Swift-based client SDK for OpenWhisk compatible with Swift 2.x and runs on iOS 9, WatchOS 2, and Darwin.
  https://github.com/openwhisk/openwhisk-podspecs
  CocoaPods Podspecs repo for ‘openwhisk-client-swift’.
  https://github.com/openwhisk/openwhisk-sdk-docker
  This is an SDK that shows how to create “Black box” Docker containers that can run Action (code).
- Package repos.:
  https://github.com/openwhisk/openwhisk-package-pushnotifications
  In-progress, Push notifications to registered devices.
  https://github.com/openwhisk/openwhisk-package-twilio
  In-progress, Integration with Twilio.
  https://github.com/openwhisk/openwhisk-package-jira
  In-progress, Integration with JIRA events.
  https://github.com/openwhisk/openwhisk-package-rss
  Integration with RSS feeds.
  https://github.com/openwhisk/openwhisk-package-kafka
  New, In-progress, Integration with Kafka
  https://github.com/openwhisk/openwhisk-slackbot-poc
  In-progress, deploy a Slackbot with the capability to run OpenWhisk actions
- Ecosystem repos.:
  https://github.com/openwhisk/openwhisk-tutorial
  Place to submit interactive tutorials for OpenWhisk, its CLI and packages. Currently, contains Javascript-based tutorial for learning the OpenWhisk CLI.
  https://github.com/openwhisk/openwhisk-vscode
  This is a prototype extension for Visual Studio Code that enables complete round trip cycles for authoring OpenWhisk actions inside the editor.
- API Gateway Framework repositories:
  There are existing discussions between IBM and Adobe about creating a comprehensive API Gateway Framework that can support community contributions. We plan to move these discussions into the Apache community and invite participation in shaping this framework to ensure the best possible solution for Serverless. At this time, the existing Adobe API Gateway provides a valuable set of modularized components that will be part of this framework and the initial submission:
  https://github.com/adobe-apiplatform/apigateway
  The main API Gateway repository containing basic configuration files and a Dockerfile to build all modules into a single container.
  Under this repository, you will find complete and conformant code modules for the following functions:
  - Request Validation (e.g., OAuth, API-KEY) and tracking,
  - Configuration syncing with multiple Cloud storage solutions,
  - API Request Caching and Mgmt.,
  - Asynchronous logging (API traffic),
  - ZeroMQ adapter with logger,
  - NGINX extensions (i.e., AWS SDK)
  - HMAC support for Lua (multiple algorithms, via OpenSSL)
    During the incubation, this code will likely be restructured to accommodate additional code from other sources as agreed to by Apache and the PPMC.

Source and Intellectual Property Submission Plan

External Dependencies

The OpenWhisk project code, documentation, samples (for all repositories) have been fully authored under the Apache 2 license with a comprehensive CLA requirements enforced for all committers from its inception. The code has been fully screened and evaluated to assure its code consists of original contributions not encumbered by any license that would be incompatible with Apache.

openwhisk-openwhisk

This repository is the primary repository for the OpenWhisk platform; it contains the implementations for all its component services, CLI and tooling.

- tooling and runtime dependencies:
  Note: all dependencies are to latest version unless noted otherwise.
- Build and Deployment Tooling:
  ansiblev2.* : GNU GPL
  Primary Runbook (playbooks) tooling for deployment with configurations for multiple target environments (ppa:ansible/ansible). Installed by ansible.sh.
  git : GPL 2
  Command line for automation of “pulling” OpenWhisk repositories’ code from Git repos. Installed by misc.sh.
  zip : Info-ZIP (BSD style)
  Tooling for decompressing files packaged in compressed ZIP format. Installed by misc.sh.
  python-pip : MIT
  Python installer. Installed by pip.sh
  jsonschema : MIT
  Python Library. JSON schema validation. Installed by pip.sh
  argcomplete : Apache
  Python Library. Bash tab completion for ‘argparse’. Installed by pip.sh
  oracle-java8-installer : Oracle Binary Code
  Oracle Java 8 Installer (Ubuntu PPA archive), Installed by java8.sh
  software-properties-common : GNU GPL v2
  Manage your own PPAs for use with Ubuntu APT. Installed by ansible.sh
  gradle 3.0: Apache 2
  Build tool.
  gradle-wrapper.jar : Apache 2
  Gradle wrapper tool. Installed by gradle-wrapper.properties
  One-JAR : One-JAR license (BSD-style)
  package a Java application together with its dependency Jars into a single executable Jar file. Used by core/javaAction/proxy/build.gradle
  npm : Artistic License 2.0
  Node Package Manager (NPM), core/nodejs6Action/Dockerfile
  Application Services:
  docker-engine, v1.9, moving to v1.12 : Apache 2
  Runtime for Docker containers. Installed by docker.sh.
  docker-py v1.9, Apache 2
  Python API client. Installed by ansible.sh.
  ntp : NTP (BSD 3-clause)
  Network Time Protocol service started to sync. peer-computer times. Note: UTC is default for all hosts. Installed by misc.sh.
  CouchDB : Apache 2
  JSON document database. Vagrant / User installed.
  Consul v0.5.2 : Mozilla v2
  Consul Key-value data store. Installed by services/consul/Dockerfile.
  - Runtime Libraries:
    Scala v2.11 : Scala (3-clause BSD)
    Primary language for OpenWhisk. Specifically: org.scala-lang:scala-library, 2.11.6. Installed by scala.sh, (referenced by build.gradle).
    Node v0.12.14: MIT
    Node JavaScript Runtime. It also includes many NPM libraries. See core/nodejsAction/Dockerfile for a complete/current list.
    Node v6.2: MIT
    The NodeJS6 Runtime. It also includes many NPM libraries. See core/nodejs6Action/Dockerfile for a complete/current list.
    Python Runtime, v2.7 (Python Std. Library) : Python
    Python based Docker Images are used in a few places. For example, see core/ActionProxy/Dockerfile. In addition, it is referenced by the Python CLI which is being deprecated as it is being replaced by a Go language CLI.
    Java 8 JRE : Oracle
    Java Language Runtime (Oracle Java 8 JDK). Referenced by common/scala/Dockerfile, core/javaAction/Dockerfile, services/consul/.
    classpath.
    Akka 2.47 Libraries for Scala 2.11 : Apache 2
    Specifically, the following: “com.typesafe.akka:” modules are used: akka-actor, akka-slf4j, akka-http-core, akka-http-spray-json-experimental. Installed by build.gradle.
    argcomplete : Apache
    Python library. Bash tab completion for argparse. Installed by tools/ubuntu-setup/pip.sh.
    httplib : Python
    Python library. HTTP protocol client. Installed by .
    jsonschema : MIT
    Python library. Installed by tools/ubuntu-setup/pip.sh.
    spray (source) : Apache 2
    Scala libraries for building/consuming RESTful web services on top of Akka. Installed by build.gradle. Specifically but not limited to: spray-caching, spray-json, spray-can, spray-client, spray-httpx, spray-io, spray-routing.
    log4j:log4j:1.2.16
    Java logging library. Installed by build.gradle.
    org.apache.* Libraries : Apache 2
    Including: org.apache.commons.*. org.apache.zookeeper:zookeeper, org.apache.kafka:kafka-clients, org.apache.httpcomponents: httpclient. See build.gradle for current list and versions.
    Including low level HTTP transport component libraries: org.apache.http.*, org.apache.httpcomponents:
    httpclient, . See whisk/common for current list and versions.

org.apache.jute.compiler.JString
urlparse : Python
Python library for URL string parsing. Referenced by tools/cli/wskutil.py
tools/build/citool.
swagger-ui 2.1.4 : Apache 2 * atypical license text
Collection of HTML, Javascript, and CSS assets that dynamically generate documentation from a Swagger-compliant API. See core /controller/Dockerfile.
Optional Services and Tooling:
Cloudant : Apache 2
(Optional) Database service. User may connect to instance from README. CouchDB can be used otherwise.
Eclipse IDE : Eclipse Public License (EPL)
Tooling, IDE. (Optional). OpenWhisk supplies a .project and .pydevproject files for the Eclipse IDE.
emacs : Emacs GPL
Tooling, Editor. (Optional) Installs Emacs editor. Installed by emacs.sh.

- Swift3 Runtime Dependencies:
  The following Python libraries are installed in the core/swift3Action/Dockerfile:
  Python 2.7 : Python
  Python Std. Library.
  python-gevent : MIT
  Python proxy support.
  python-distribute : PSF (or ZPL)
  Supports the download, build, install, upgrade, uninstall of Python packages. See: http://pythonhosted.org/distribute. Note: this is a fork of: https://github.com/pypa/setuptools.
  python-pip : MIT
  PyPA recommended tool for installing Python packages.
  python-flask : BSD
  Python proxy support.
  clang : NCSA Open Source
  'C' Library. Apple compiler front-end for ‘C’ (LLVM back-end).
  libedit-dev : BSD (3-clause)
  Linux, BSD editline and hostry library.
  libxml2-dev : MIT
  Linux, Gnome XML library.
  libicu52 : Unicode
  Linux, Unicode support library.
  Kitura : Apache 2
  Web framework and web server that is created for web services written in Swift.
  Kitura dependencies : BSD (BSD-like)
  Linux libraries including: autoconf, libtool, libkqueue-dev, libkqueue0, libdispatch-dev, libdispatch0, libcurl4-openssl-dev, libbsd-dev.
  apple/swift-corelibs-libdispatch : Apache 2
  Enables Swift code execution on multicore hardware.

Adobe-API-Platform

Openresty - Licensed under the 2-clause BSD license - https://github.com/openresty/ngx_openresty#copyright--license
NGINX License - http://nginx.org/LICENSE
Luajit - MIT License - http://luajit.org/luajit.html
PCRE - BSD license - http://www.pcre.org/licence.txt
NAXSI: GPL - is not compiled with the Gateway API code. Instead The API Gateway project contains instructions for developers on where to get NAXSI code (under GPL)
ZeroMQ / ØMQ - Linked Dynamically in separate module
libzmq - LGPL license with SPECIAL EXCEPTION GRANTED BY COPYRIGHT HOLDERS - https://github.com/zeromq/libzmq
czmq - High Level C binding for libzmq - MPL v2 license https://github.com/zeromq/czmq

Trademarks

IBM is pursuing trademarking of the OpenWhisk name in the following jurisdictions: Canada, France, WIPO (i.e., Australia, China, CTM (EUIPO), India, Mexico, Russian Federation, Switzerland, United States of America). IBM plans to transfer all filings and trademark ownership to ASF.

Cryptography

Please note that the file https://github.com/openwhisk/openwhisk/blob/master/common/scala/src/main/scala/whisk/common/Crypt.scala makes use of the Java javax.crypto.* libraries to implement encrypt/decrypt functions. Primarily this is used to encrypt/decrypt user keys or secrets when being passed or stored between or by OpenWhisk components.

In addition, the API Gateway modules (api-gateway-hmac) relies on OpenSSL (openssl/evp.h, openssl/hmac.h).

Required Resources

Resources that infrastructure will be asked to supply for this project.

Over the course of the incubator we would like to develop staging and playground server environments for testing and developer experience. The following environment would be desirable for an initial staging (and separate playground):

- CI Test Cluster requirements:
  3 VMs, Catalog (CouchDB/Cloudant), Router (Nginx), Registry

2 VMs, Master (Controller + Consul), Message Bus (Kafka)
10 VMs, Invokers
Each VM assumes 4 CPUs, 8GB Memory, 80GB additional storage

- Mechanics:
  Scripts that invoke Ansible playbooks for build, deploy (run) and clean are provided.
  The various architectural components are started via Docker containers (either natively, within a single Vagrant VM, or across multiple, designated VM roles) using user configured (or defaulted) endpoints and (guest) authorization credentials.
  In addition, the user/developer may choose to use the default ephemeral CouchDB (via Docker container) for the OpenWhisk catalog or switch to use a native CouchDB or a remote Cloudant database.

In addition, we would like to host a VM with a Node.js server that provides Command Line Tutorials, along with demo samples.

Mailing lists

Initially, we would start with the following recommended initial podling mailing lists:

private@openwhisk.incubator.apache.org,
dev@{podling}.incubator.apache.org

We would add more as we transition off exiting mailings lists and through the course of incubation.

Git Repository

As a community we would like to keep the master repository as well as issue tracking on [GitHub]. We will be working closely with ASF Infra. team to implement all the required pieces like ensure to send push and issue notifications through ASF controlled mailing lists. During incubation we will work closely with Infra to support [GitHub] master repositories. We also understand that we have to support a way of providing patches, which does not require a [GitHub] account for contributors who are not willing or not able abide by [GitHub]'s terms and conditions. It is our understanding that this approach has been signed off by Greg Stein, ASF's Infrastructure Administrator.
gstein sez: the podling can only graduate within an approved repository system. The IPMC may have a differing opinion, but from an Infra perspective: the OpenWhisk podling can continue with their usage of a [GitHub] repository, but faces a clear obstacle: [GitHub] "as master [as allowed by the Foundation]" must be approved and working before the graduation, or they must migrate their primary to the Foundation's Git repository (at git-wip) before they graduate.

If we need to adapt our repo. paths to conform to Apache guidelines (and perhaps necessitated by a move the the Apache named repo.) It is conventional to use all lower case, dash-separated ⛔ repository names. The repository should be prefixed with incubator and later renamed assuming the project is promoted to a TLP.

If we need to move the project codebase from its existing GitHub repo. as part of incubation, we would like to preserve the directory names as they appear today and adopt the "apache" as part of the URI path as we have seen other projects adopt.

This would mean all existing repositories which are now of the form:

- https://github.com/openwhisk/openwhisk
- https://github.com/openwhisk/openwhisk-catalog
- https://githun.com/openwhisk/openwhisk-package-rss
- etc.

would now take the form:

- https://github.com/apache/openwhisk/openwhisk
- https://github.com/apache/openwhisk/openwhisk-catalog
- https://githun.com/apache/openwhisk/openwhisk-package-rss
- and so on ...

Issue Tracking

We would like to explore the possibility of continuing to use GitHub issue tracking (as project milestones, epics and features are all nicely tracked via ZenHub boards) as we understand that this may now be possible. We will provide any linkage or support for JIRA issue tracking if that is required in order to track any "pull" requests within GitHub.

Other Resources

We would like to preserve our existing automated TravisCI automated testing from GitHub. The project uses a continuous CD/CI process currently that we would like to continue to support via multiple stages that run progressive stress and performance tests that are also automated.

Initial Committers

The following is the proposed list of initial committers, email address [, GitHub ID)]:

- Bertrand Delacretaz, bdelacretaz@apache.org, bdelacretaz
- Carlos Santana, csantana@us.ibm.com, csantanapr
- Carsten Ziegeler, cziegeler@apache.org, cziegeler
- Chetan Mehrotra, chetanm@adobe.com, chetanmeh
- Christian Bickel, CBICKEL@de.ibm.com, christianbickel

- Daisy Guo, guoyingc@cn.ibm.com, daisy-ycguo
- David Liu, david.liu@cn.ibm.com, lzbj
- Dragos Dascalita Haut, ddascal@adobe.com, ddragosd
- Jeremias Werner, JEREWERN@de.ibm.com, jeremiaswerner
- Markus Thommes, markus.thoemmes@de.ibm.com, markusthoemmes
- Matt Rutkowski, mrutkows@us.ibm.com, mrutkows
- Nicholas Speeter, nwspeete@us.ibm.com, nwspeete-ibm
- Paul Castro, castrop@us.ibm.com, paulcastro
- Perry Cheng, perry@us.ibm.com, perryibm
- Philippe Sutor, psuter@us.ibm.com, psutor
- Rodric Rabbah, rabbah@us.ibm.com, rabbah
- Sergio Fernández, wikier@apache.org, wikier
- Stephen Fink, sjfink@us.ibm.com, sjfink
- Tony Ffrench, tffrench@us.ibm.com, tonyfrench
- Vincent Hou, shou@us.ibm.com, houshengbo
- Edward J. Yoon, edward.yoon@samsung.com, edwardyoon

Although this list of initial committers appears long, OpenWhisk is a complete platform which consists of many services supporting many environments, programming languages and integrations. This diversity in needs is reflected by the size of the initial committers group. OpenWhisk also supports an end user ecosystem including CLI, Tooling, Package Catalog, “curated” Packages, samples, etc. along with the intention of tying in API gateway (e.g., OpenAPI) and other event source integrations.

We hope to add many more committers who provide expertise and the various areas OpenWhisk uses to efficiently provide an exceptional Serverless platform with compelling content.

Affiliations

Additional TBD during the proposal process

Sponsors

Additional TBD during the proposal process.

Sponsoring Entity

OpenWhisk would ask that the Apache Incubator be the sponsor.