Towards Sustainable Ecosystem for Cloud Functions

Authors:
Yessica Bogado - Itaipu Technological Park
Walter Benitez - Itaipu Technological Park
Josef Spillner - ZHAW School of Engineering
Fabio López-Pires - Itaipu Technological Park

CONTENT

1. Challenges
2. Ecosystem

   Analysis, obstacles.

3. Proposition
4. Sustainable Ecosystem Elements
   Marketplaces, Converters, Deployers, Execution Environments

5. Proof of Concepts
   Function Hub

6. Conclusion

1.

CHALLENGES

Technology-Specific Exchanges

But...

Which future exchanges may emerge for new technologies such as **cloud functions**?

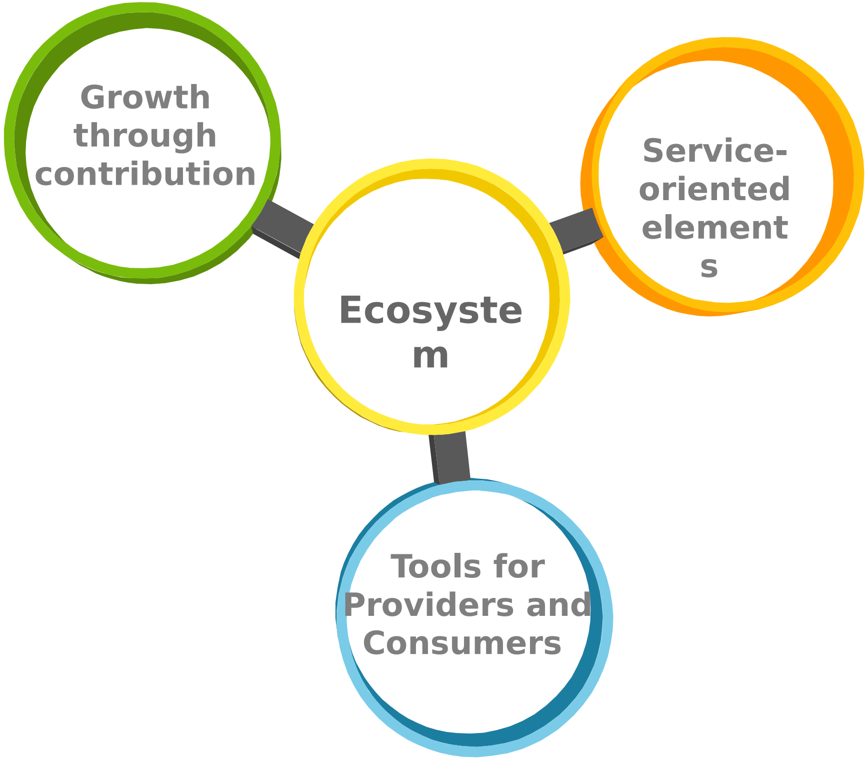
How these future exchanges will **look like**?

Who operates such exchange in a **sustainable** way?

2.

ECOSYSTEM

What is an Ecosystem?



Growth through contribution

Service-oriented elements

Ecosystem

Tools for Providers and Consumers

Growth Ecosystem

The dependent products grow slower with logarithmic relation compared than the independent products.

Ecosystem Obstacles

1. Single Commercial Owners

2. Concentration of providers in ecosystems

Cloud Function Ecosystem

In Serverless architecture, cloud providers have complete management over the environment in which functions run.

3.

PROPOSITION

Proposition

Establish Sustainable Ecosystems for heterogeneous application development artefacts which can be customised for arbitrary domains.

How?

Decentralisation

Guarantees that in the worst case the system will continue to function in reduce form.

Abstraction

Converting formats and protocols.

4.

SUSTAINABLE ECOSYSTEM ELEMENTS

Marketplaces

Converters

Deployers

Execution Environments

Marketplaces

Environment where developers could interact with the platform ecosystem in a way that allows them to create, share and trade tools.

Enabling users to deploy, scale and create functions more **easily** and **efficiently**.

GitHub

docker

awsmarketplace

Marketplaces

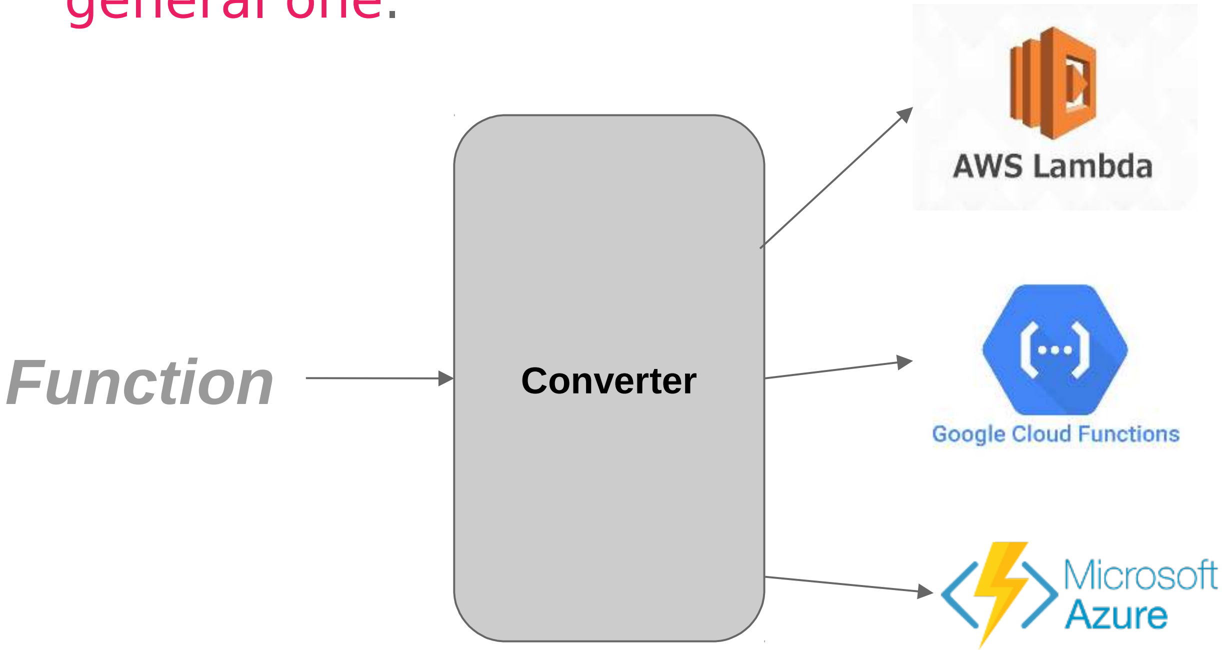
Converters

Deployers

Execution Environments

Converters

Users are forced to create functions for specific cloud providers instead of a general one.



Function

Converter

AWS Lambda

Google Cloud Functions

Microsoft Azure

Marketplaces

Converters

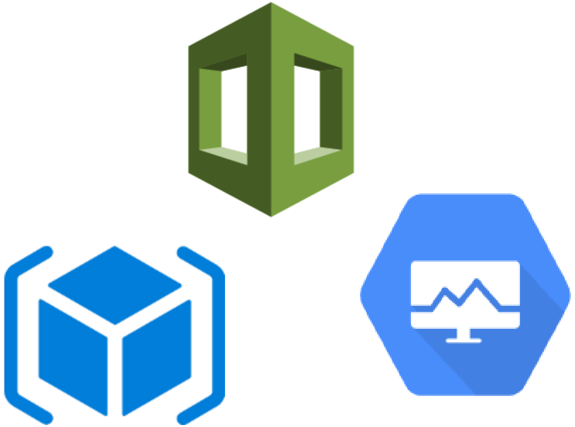
Deployers

Execution Environments

Deployers

The users need a flexible tool that allow them to deploy their functions on multiple cloud environments.

Specific platform



Multi-Tenancy

serverless

Marketplaces

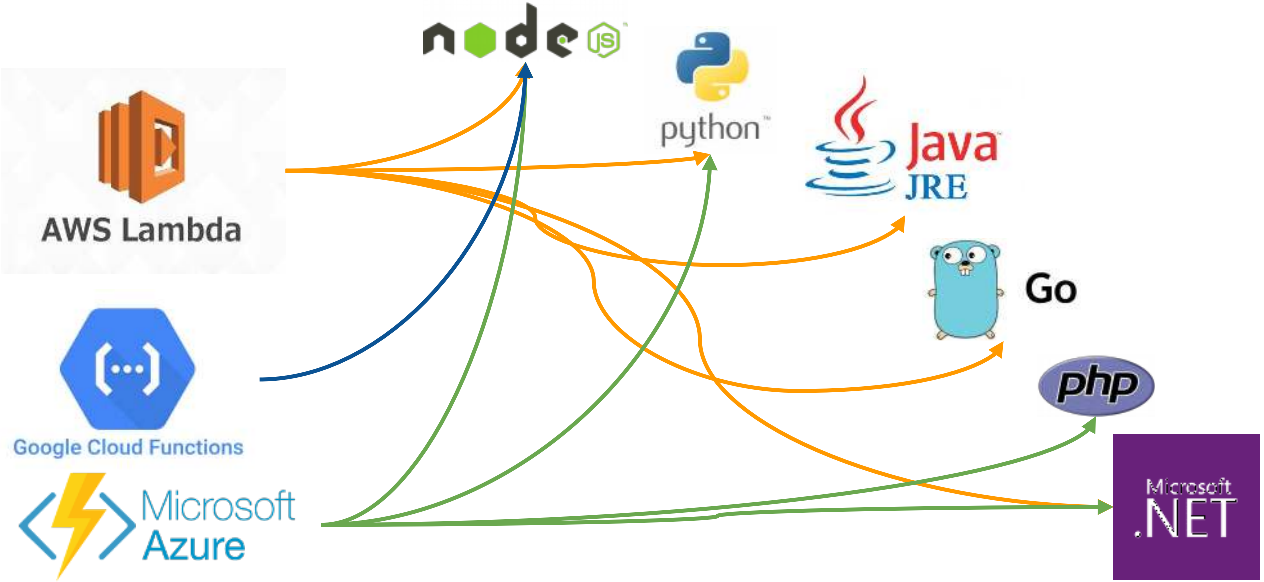
Converters

Deployers

Execution Environments

Execution Environments

Each cloud provider focuses its environment in accordance to an aimed developer group or their specific infrastructure.



Similar Ecosystem

5.

PROOF OF CONCEPTS

Marketplaces

Converters

Deployers

Execution Environments

FunctionHub

Marketplaces

Converters

Deployers

Execution Environments

Marketplaces

FunctionHub allows free exchange of functions between users and generates the required environments for a serverless market to proliferate.

Decentralisation	Abstraction
Extensible Mesaging and Presence Protocol (XMPP)	Snafu

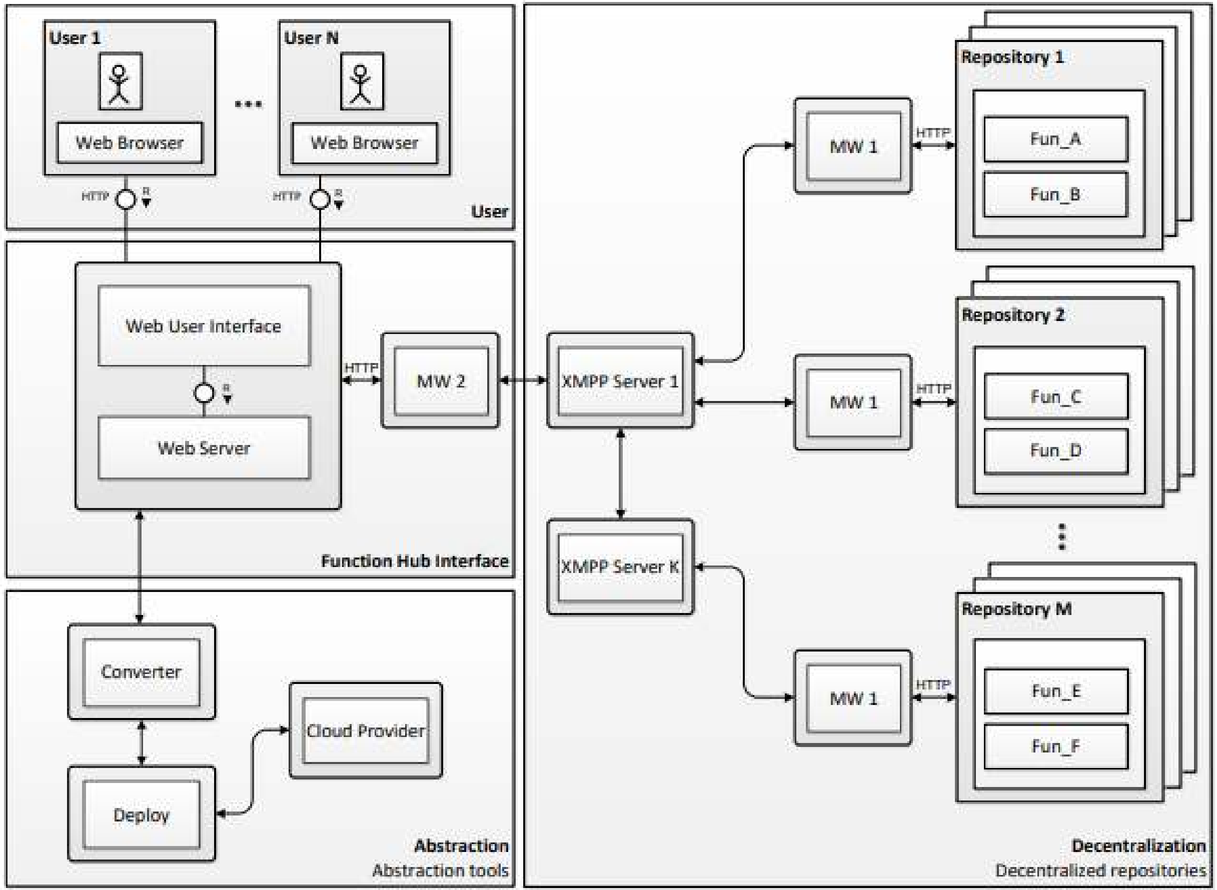
Marketplaces

Converters

Deployers

Execution Environments

Marketplaces



Marketplaces

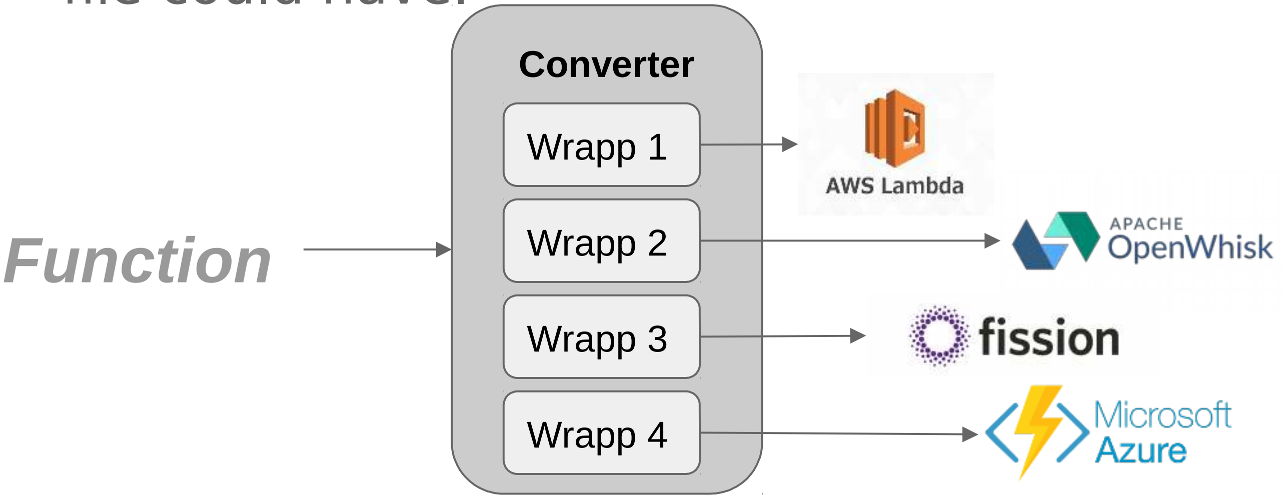
Converters

Deployers

Execution Environments

Converters

As a early prototype, a converter of Python functions was developed to add wrappers for different modules that the file could have.



Function

Converter

Wrapp 1

Wrapp 2

Wrapp 3

Wrapp 4

Marketplaces

Converters

Deployers

Execution Environments

Deployers

Snafu give users the option to upload their functions from their repositories to the *Function Hub* ecosystem.	
	For deploy functions from *Function Hub* to a private cloud provider is intended to use the Serverless Framework.

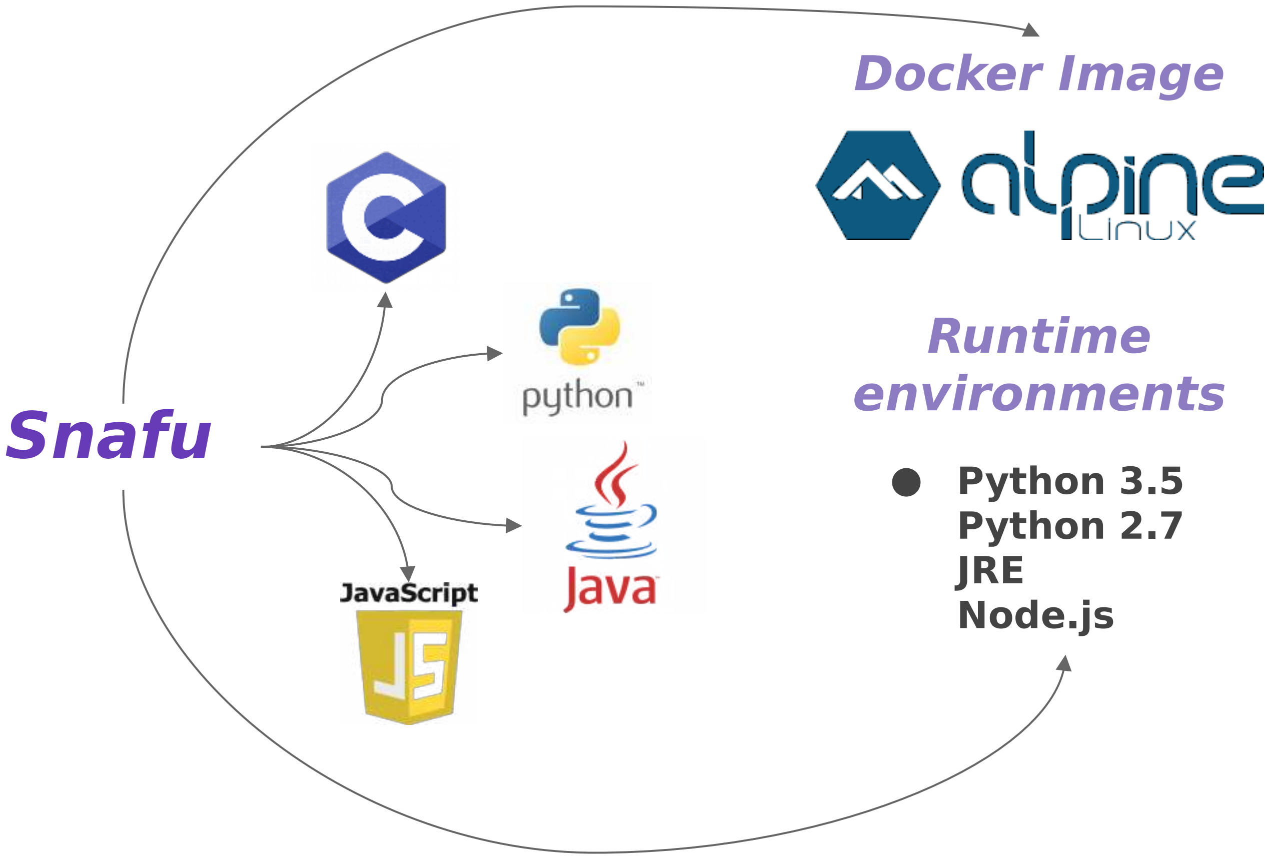
Marketplaces

Converters

Deployers

Execution Environments

Execution Environments



Snafu

Docker Image

Runtime environments

- **Python 3.5**
  Python 2.7
  JRE
  Node.js

6.

CONCLUSION

Conclusion

The rapid growth of Serverless Computing creates a need for an ecosystem in order to bring users necessary tools for a **fast** and **cheap** deployment of their software.

It is needed properties like **decentralisation** and **abstraction** that allows users to create applications that interact with a diverse cloud ecosystem and take advantage of this diversity according to their needs.

THANKS

yessica.bogado@pti.org.py

walter.benitez@pti.org.py

fabio.lopez@pti.org.py

josef.spillner@zhaw.ch