Expedient: A Pluggable Non-Intrusive Exploratory Control Framework for GENI

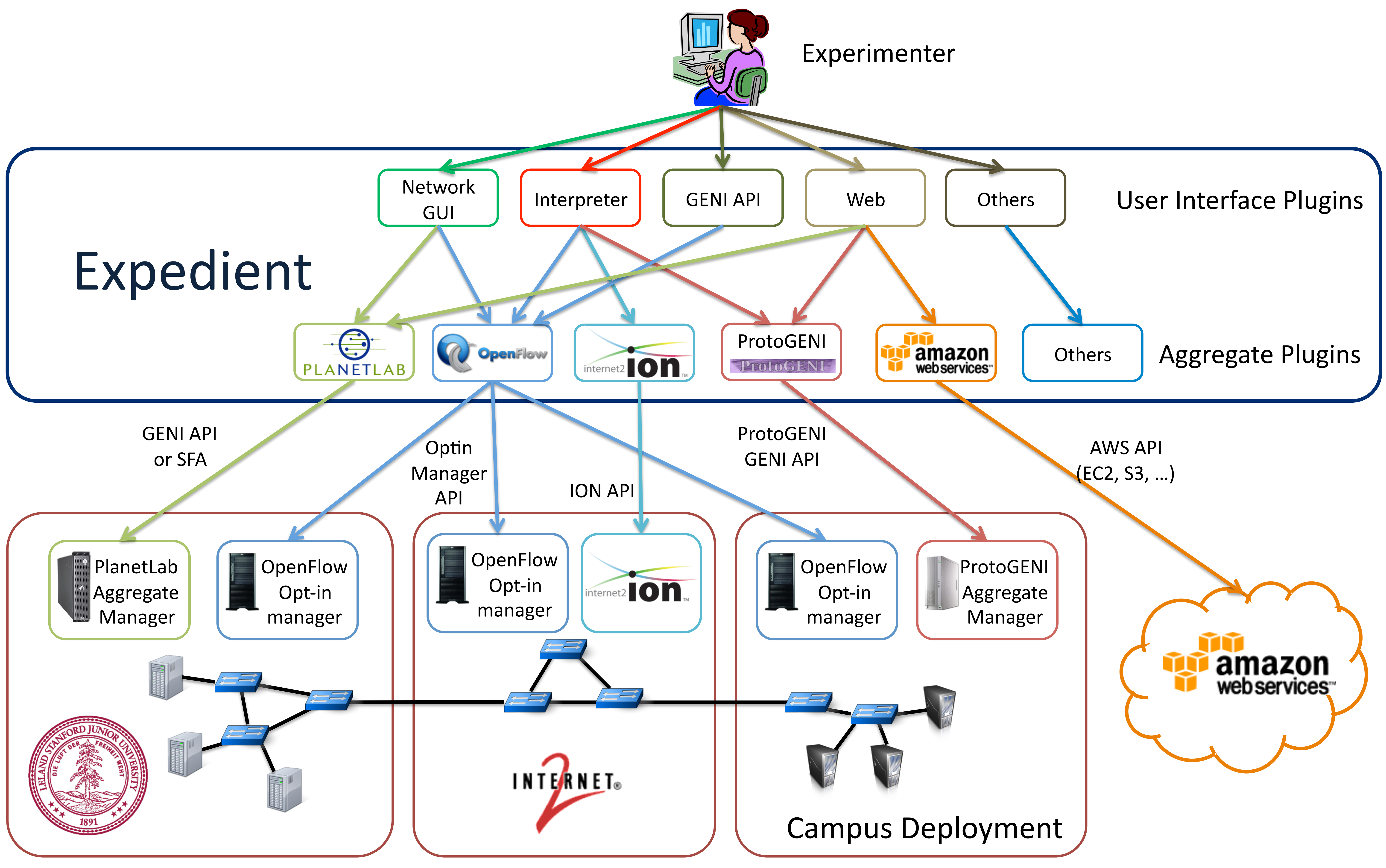
Jad Naous, Peyman Kazemian, Rob Sherwood,
Guido Appenzeller, Guru Parulkar, Nick Mckeown

Makes GENI easier for **Developers**

1. Expedient **does not interfere** with how resource developers design their resources. They do not have to compromise in order to conform to an ill-fitting API.
2. Expedient **simplifies authentication and authorization**. It acts as a gateway for user transactions so developers do not have to.
3. Expedient provides a **powerful platform** for connecting new types of resources and designing rich user interfaces. Expedient uses **Python** and **Django** as a Web framework.

Makes GENI easier for **Users**

1. Expedient is a **Web application**. It can be accessed from anywhere anytime, does not require downloads, and does not need to be updated and maintained.
2. Expedient provides **rich and powerful user interfaces** that are tailored to the resources they want to use.
3. Users **do not need to manage and maintain certificates and credentials** for each type of aggregate. Expedient does it for them.



Provide a complete platform

Integrating with aggregates:

We plan on providing aggregate plugins for **PlanetLab**, **ProtoGENI**, **OpenFlow**, **Amazon Web Services**, and Internet2's **ION** service by GEC9. Today, we have PlanetLab and OpenFlow under alpha testing.

User Interfaces:

User interfaces must be tailored to the resources they are trying to reserve. We plan on providing user interface plugins that allow reservations using an interactive GUI, an interpreter, and simple HTML pages.

Contributing:

We invite you to develop aggregate or user interface plugins for Expedient. It is simple and easy to connect existing resources using the protocol they already expose.

Developing Aggregate Plugins

Step 1. Decide on your protocol and API.

You will need to decide what transport you will use and what remote calls you will be making.

Step 2. Design classes to represent your resources in the Expedient database.

Expedient provides a number of base classes with sane default functions you can extend to simplify your job.

Step 3. Write up a little bit of HTML.

This HTML will mainly be used to specify customized settings for your aggregate when it is added to Expedient or to a slice.

Step 4. Write tests.

Expedient uses Django's test framework to make sure that your classes and pages work as expected.

Step 5. Deploy!

For more information visit http://yuba.stanford.edu/~jnaous/expedient/