CPRE 492 WEEKLY REPORT 19

Project Molecule

15 – 21 February 2017

May1739
may1739@iastate.edu

Dr. Arun Somani

Ryan Wade – Team Leader

Nathan Volkert – Communications Lead

Daniel Griffen – Key Concept Holder

Alex Berns – Webmaster & Scribe

1 CONTENTS

2 Weekly Summary

We each worked on individual parts. The next big step is the integration phase itself, which we discussed the plan for that in advisor meeting.

3 Past week accomplishments

All Members:

- Discuss Integration steps for coming week

Ryan Wade:

- Fixed Bugs in UI Library
- Ported UI to Rust
- Integration Planning for Atomic Layer (See 5.3)

Nathan Volkert:

- Worked on app permissions
- Created app actions (Install, uninstall, run, stop)

Daniel Griffen:

- Integrated molecule-net and molecule-common
- Created trait abstraction across backend implementations

Alex Berns:

- Form Builder
    - render view with tabs, and ability to add new scenes
    - Flat view now properly render
    - Refactored Code
    - Started to add components

4 Individual contributions

NAME	Hours	Semester Total	Cumulative
Ryan Wade	18	81	201
Nathan Volkert	20	72	174
Daniel Griffen	15	72	206
Alex Berns	17	81	180

5 COMMENTS AND EXTENDED DISCUSSION

5.1 APP ACTIONS

Apps have specific actions that can be called unique to the apps

All apps must have four basic actions

- Install – install the app on a node
- Uninstall – Removes an app from a node
- Run – Allows the app functionality to be called
- Stop – Stops app from running on node

All apps need to be registered on the system in order to be installed and run

- This is done in the form of adding the manifest files to the system. On install the app folders are added to the system as well.

5.2 BASIC PERMISSIONS

Permissions are stored as key-value pairs as Action-Nodes with access

Node A will want to make use of an action available on Node B.

The permission handler first checks if A is whitelisted on the Action of the App.

Then the permission handler checks if B is able to run the app via its permissions.

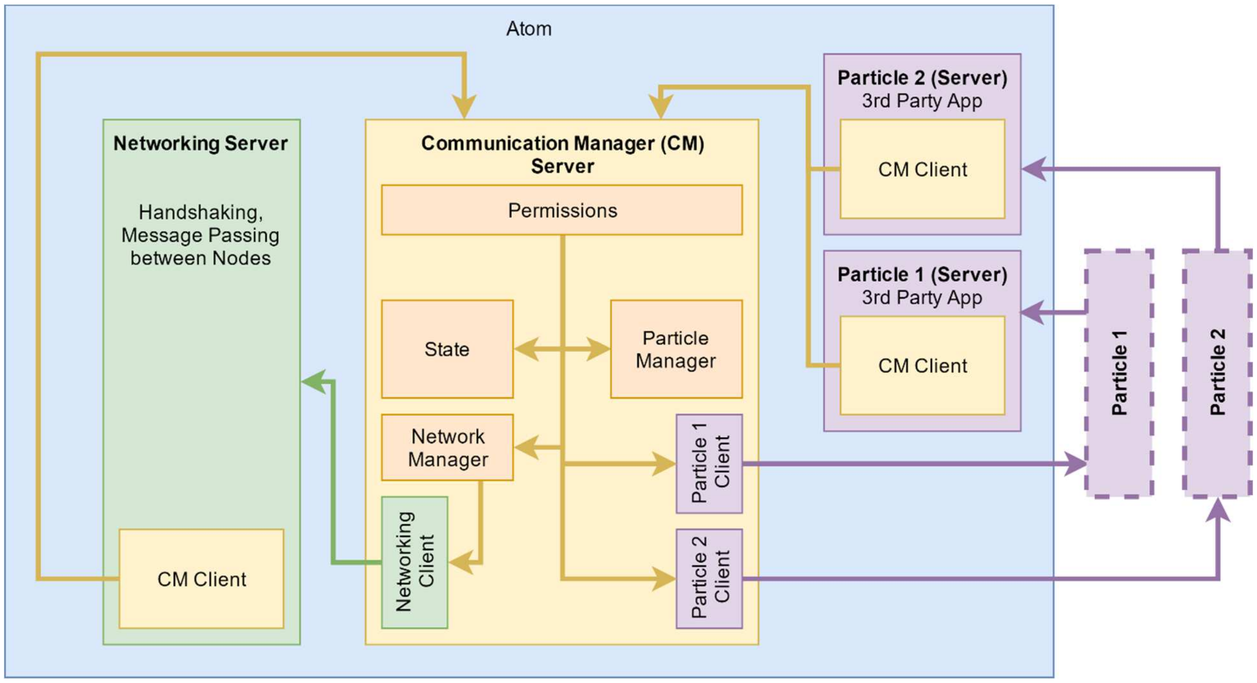
Lastly the handler ensures the action is installed on B and the transaction can complete.

On success, message transmits, on error permissions are denied.

Handler(Message, State) -> Result< Message, ()>

5.3 Communication Manager (Atomic Layer Event Loop)

This week we worked on an integration plan for Messaging Services (Networking & Socket Communication) and the Core Atomic Layer Features. We determined the following threaded Architecture:



The Atom is comprised of two threads handling Network communication and an event handling loop. Each atom also has a thread to handle incoming messages for each particle and forwards them to the Communication manager.

6 Plan for coming week

Alex: Form Builder will be finished.

Nat: Permission integration with Atomic Layer, better documentation and testing

Dan: Unix Sockets, documentation of sockets and layers

Ryan: Communication Manager Event Loop, State & UI Particle

7 SUMMARY OF WEEKLY ADVISOR MEETING

7.1 COVER LAST WEEK GOALS

Form Builder is behind. Also added the need for exporting UIs

Permissions is handling messages and manifest files for A and B. Also, checks that the app is installed that will handle the message.

Each app has lifecycle methods.

Streams are demo-able.

Network layer is not demo-able. Implementation was determined.

Migration of Node server to rust is going well.

7.2 NOTES

Integration is our biggest issue.

Dr. Somani leaves on the 27 at lunch. So demo must be the 26 or morning of 27

CC Dr. Somani when contacting Dave.

7.3 NEXT WEEK GOALS

Start integration steps

Integration Parts

- UI Builder Rust
- Atomic Layer communication with other atomic layer
- Atomic Layer to run permissions
- UI Particle in system