Pluggable modules and actor model

Status	NOT STARTED
Stakeholders	
Outcome	
Due date	
Owner	

Background

To support a various number of use cases Iroha2 must implement logic to plug additional logic as an actor. The design should allow an arbitrary actor to receive messages from other actors and share messages about his work.

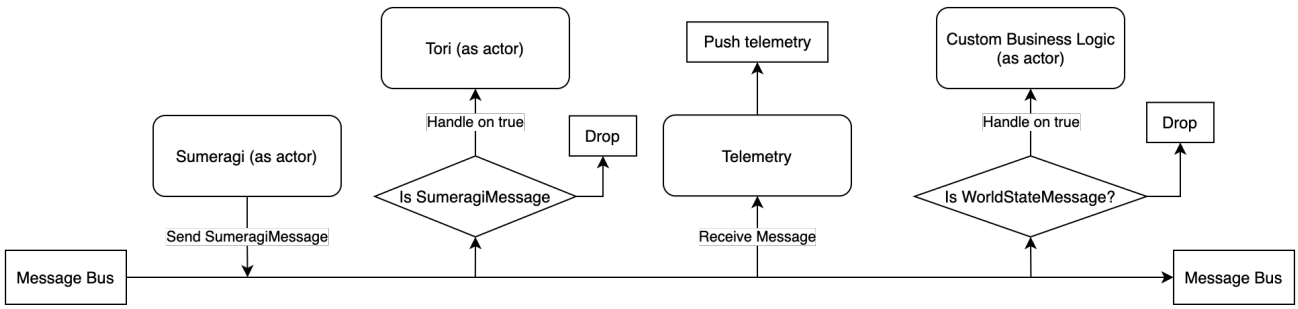
Problem

The current runtime design is implemented using mpsc. The single receiver channel architecture turns into a chaos of strict dependencies and creates constraints for extending logic with new modules.

Solution

Rework core architecture with actor model and single message bus. Bus receives all messages during execution of actors and broadcast them to others. Message trait should allow identify nature of message to filter it. To prevent any blocking problems, messages must reference an immutable value or a cloned value.

According to the requirement of supporting different use cases, actors should be implemented as compile-time features (at least) or as pluggable extensions (point to discuss and future research).



Decisions

Alternatives

1. Use available actor frameworks such as Actix or different

Concerns

1. Tradeoff between extensibility and overhead with maintenance global bus and filter messages

Assumptions

1. Filtering messages on message bus layer against skipping on actors
2. Research pros and cons of actors as separate binaries with external API to extend pluggability

Risks

Additional Information

https://github.com/jonhoo/bus
Simple implementation of a lock-free, bounded, single-producer, multi-consumer, broadcast channel

https://github.com/actix/actix
Popular actor framework for Rust

https://video.ethz.ch/events/2017/rust/e2d74a7b-7cfd-4bd3-8ce5-23c99ad775e2.html
Type-safe & high-perf distributed actor systems with Rust by Eickhoff, Anselm