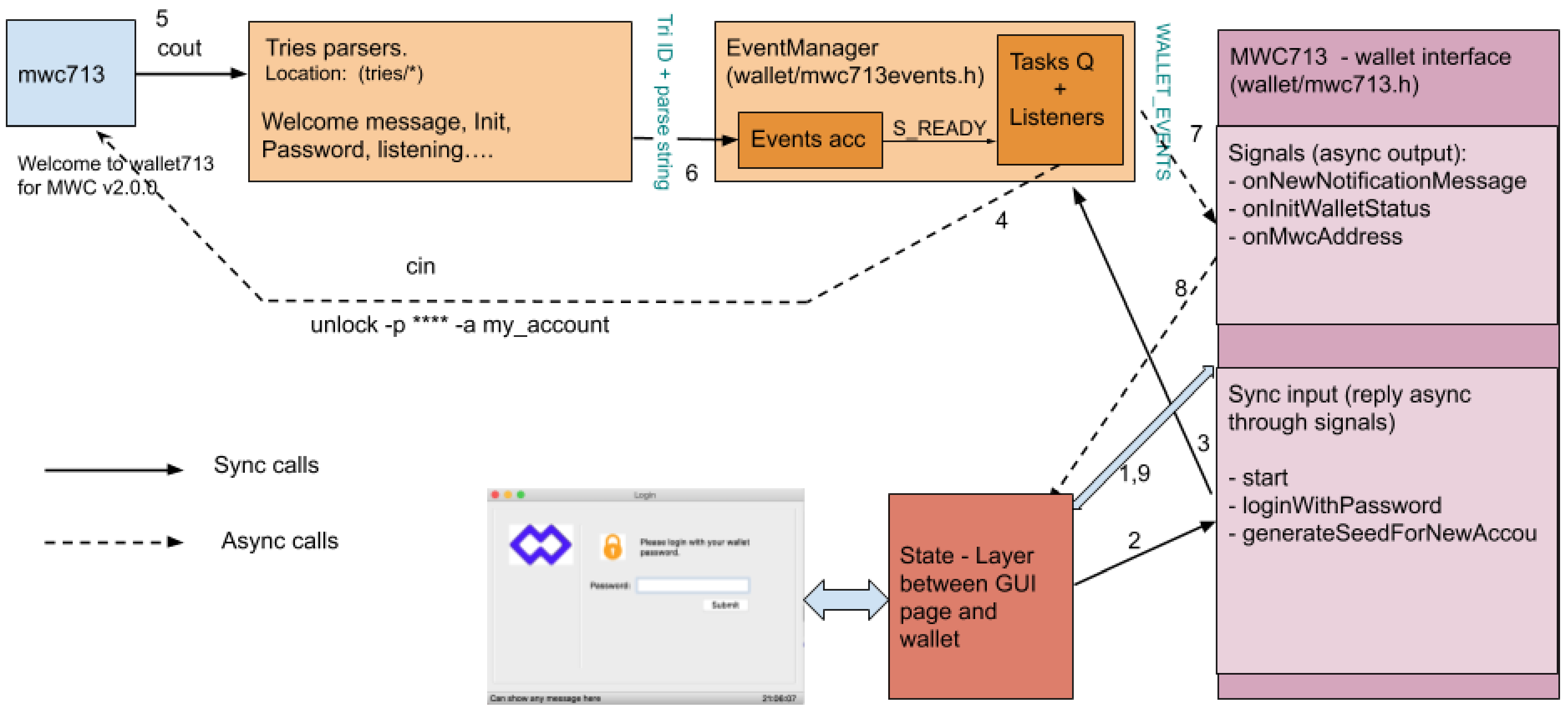
Mwc wallet design ans a workflow example



mwc713 - mwc713 process that runs in a separate process. Later we might have a standalone node as a process. Currently node not in the picture because it is not important for overall design.

MWC713 - interface between app and mwc713 process. MWC713 owns the process, tries, event manager. For testing there is a mock wallet that has the same interface.

Task Q + Listeners:
Task Queue - It is bunch of 'Tasks' that represent what user can input into mwc713 and processing of mwc713 respond.
Listeners - Set of special tasks that processing all input request. Example: "Error: XXXXXX" or "listening for MWC MQ, your address XXXXX"

State - Object that represent this app 'mwcwallet' state machine. Normally one state is assigned to a single page or logic decision (wallet init). State is a layer that separate UI from backed wallet (mwc713)

The workflow for a single action: "login with password":

1. GUI provide input 'password' to State, State call 'connect' to establish async message delivery from MWC713 (backed mwc713 wallet inteface)
   State activates 'waiting' GUI
2. State call wallet method, for example loginWithPassword(password)
3. MWC713 enrich loginWithPassword call with last used account and create task TaskUnlock(password, account). Task put into the task Queue at event manager.
4. Task it is first in the Q, Event manager is ready to execute it. Task generate input command for wmc713 'unlock -p **** -a my_account'.
5. mwc713 takes simulated input in processing. Mwc713 console output is redirected to Tries parsers. Tries parsers set is static and we expect to have the full set that will be able to parse any reasonable output from the wallet. Because all parsers are active, mwc713 can write outputs that we expecting almost in any order. If succeed, parsers write ID + result to Event manager.
6. Event manager first process income requests with listeners. Then put it into the 'Events Accumulator'.
   Event manager wait for S_READY event. It is response to "\nwallet713> ". S_READY mean that waller finish processing the task and we can check the results.
7. Task processing what events Accumulator has and make async call (emit signal) with 'unlock' command response.
8. State receive the signal in GUI thread. Response has the 'unlock' response that said if password was wrong.
9. State call 'disconnect' from async channel.
   State deactivates 'waiting' GUI.
   State updates GUI with result and user can see the login results.