Cross-Control Frameworks Federation

Sangjin Jeong, Myung-Ki Shin, Ki-Hyuk Nam

{sjjeong, mkshin, nam}@etri.re.kr

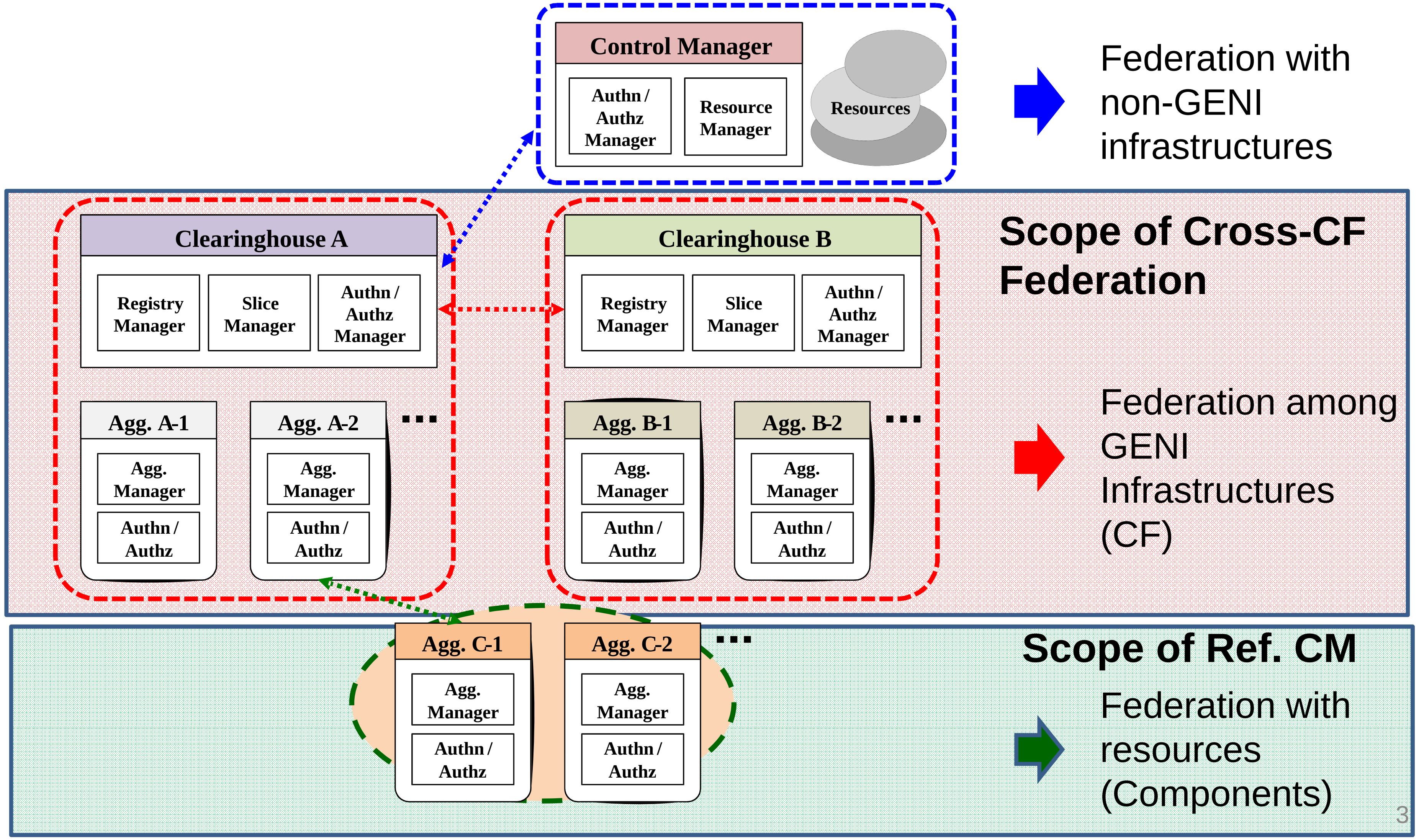
Nov 2, 2010

ProtoGENI Cluster meeting @ GEC9

Objective

- Discuss the federation among different control frameworks

GENI federation scenarios



Problem description

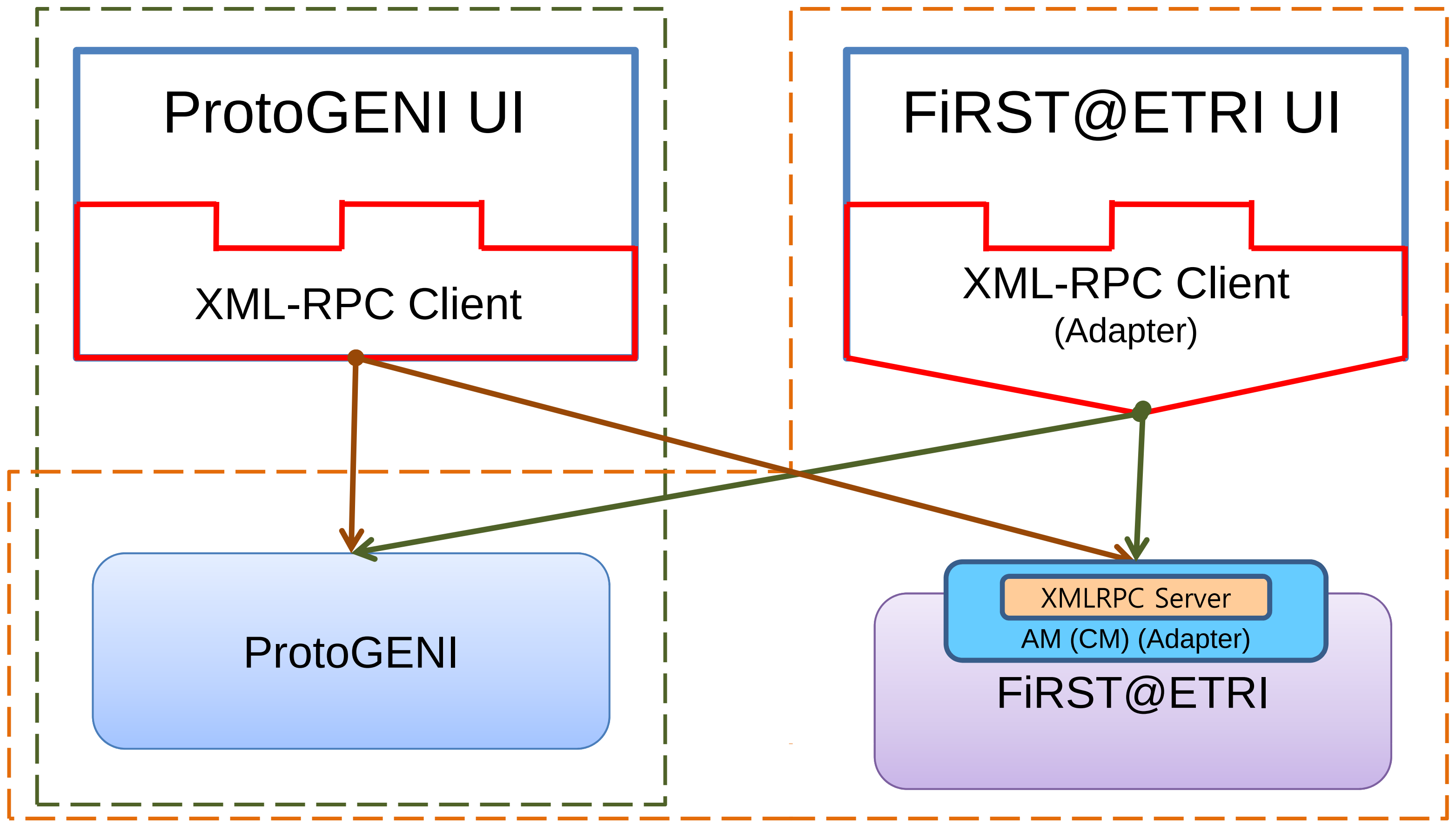
- Classification of federation problems
  - Different identity/authority management
    - Identity allocation/authorization policy/mechanism
    - Ex) GID, ABAC (SFA 2.0)
  - Different control procedures
    - Control flows and interfaces/APIs
    - Ex) GENI AM/CM/Slice APIs
  - Different resources and experiments description
    - Resource description schemes (syntax, context, entity, …)
    - Description of experiments, services, experimental results
    - Ex) Rspec

Cross-CF federation approaches

- Adapter-based federation
  - Short-term approach
  - Modification on CF entities
  - Simple but not scalable

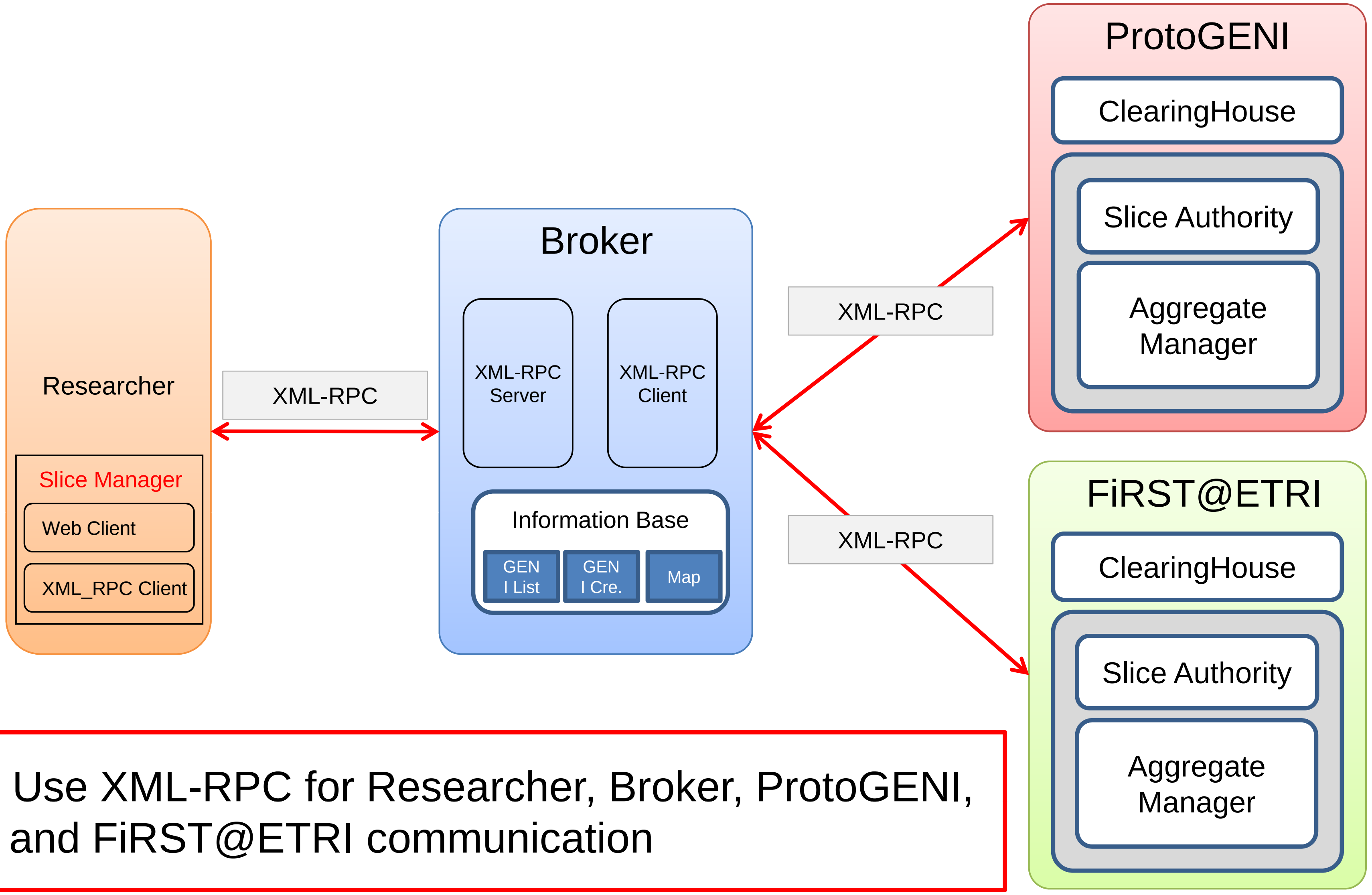
- Broker (or Arbiter)-based federation
  - Long-tem approach
  - No (minimum) modification on CF entities
  - Complex (credential issue) but scalable

Adapter-based federation



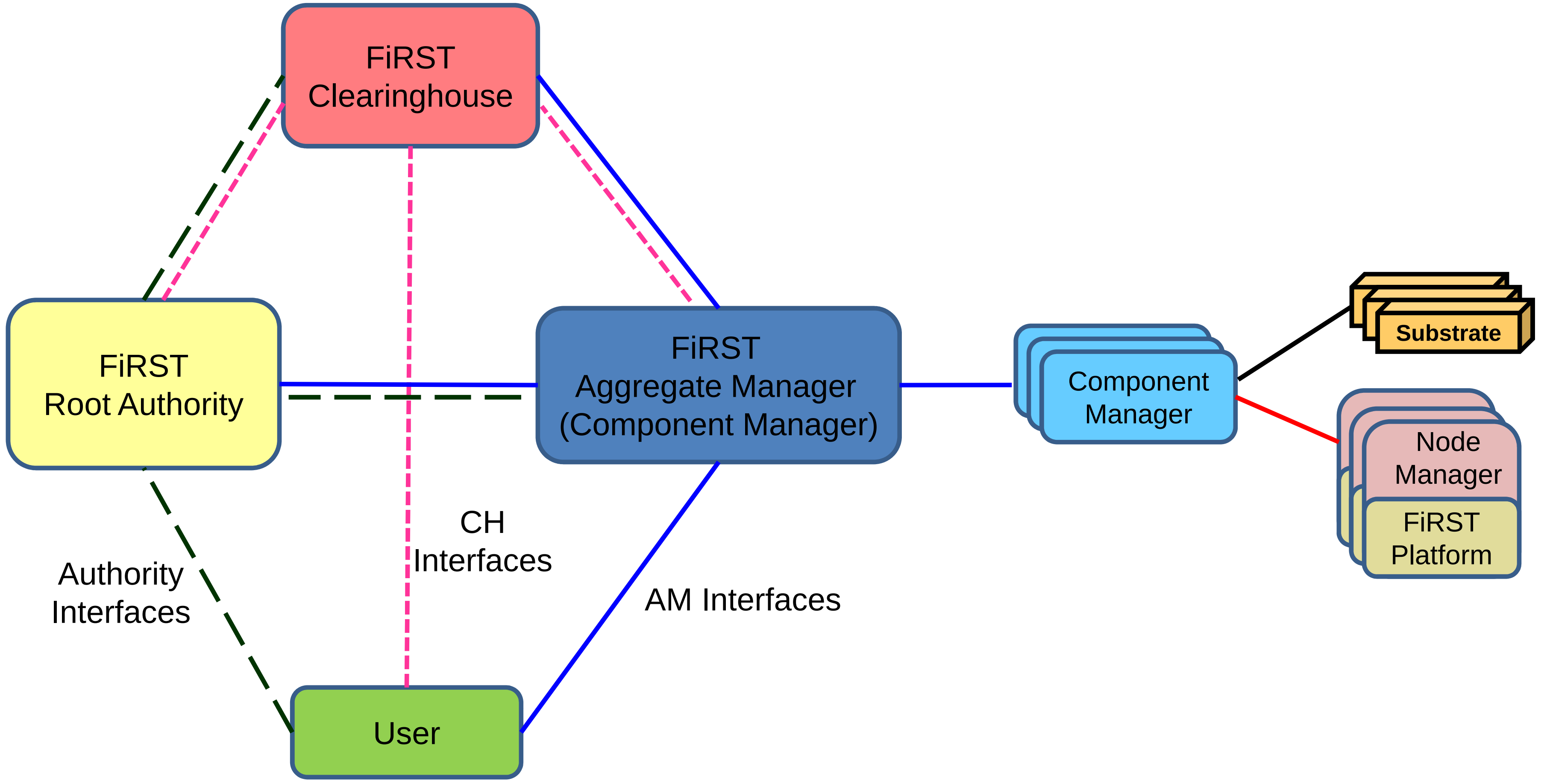
* User/AM(CM) should be registered to both ProtoGENI and FiRST@ETRI

Broker-based federation

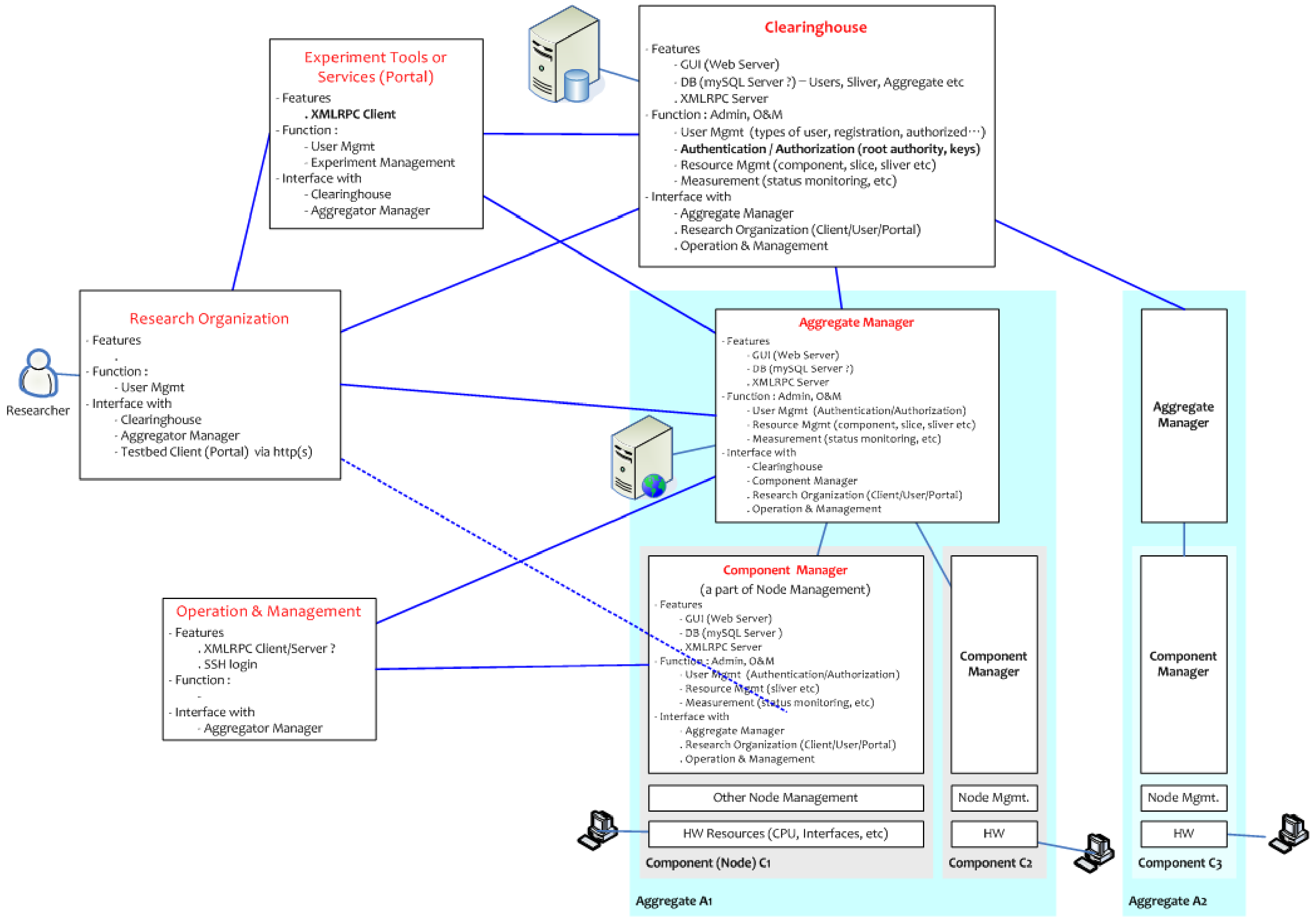


- Use XML-RPC for Researcher, Broker, ProtoGENI, and FiRST@ETRI communication

FiRST@ETRI CF entities

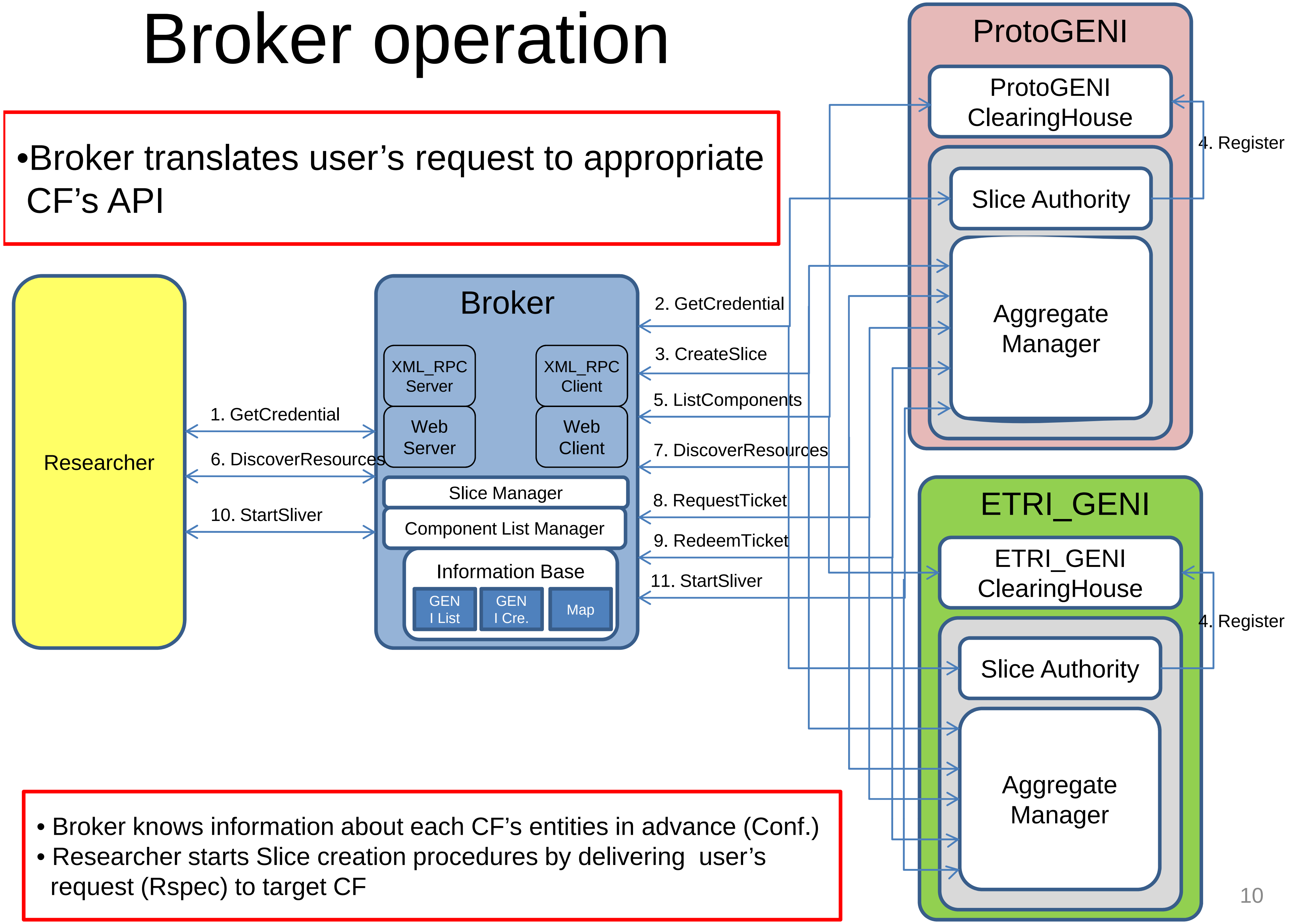


FiRST@ETRI CF entities



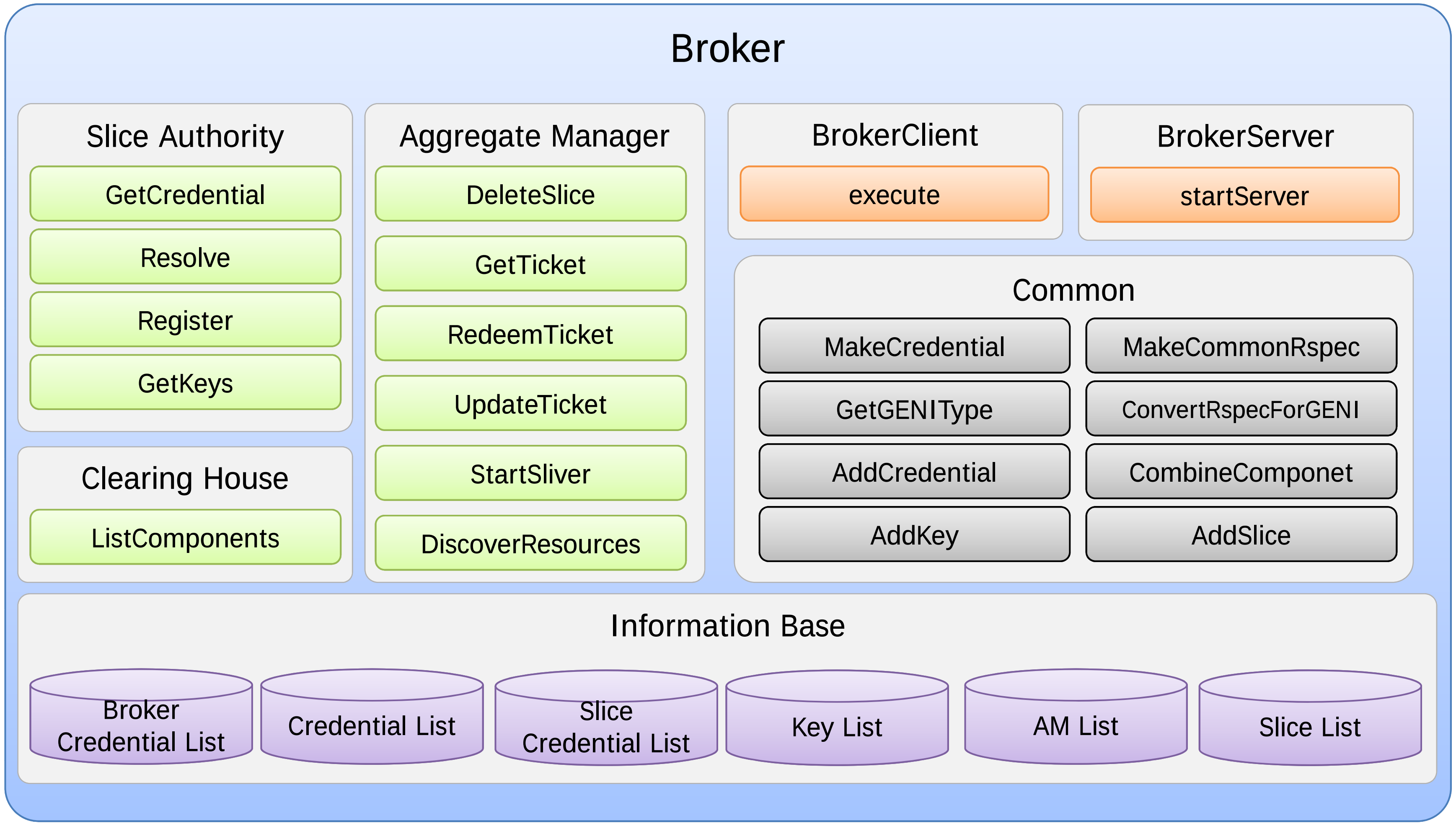
Broker operation

•Broker translates user's request to appropriate CF's API



• Broker knows information about each CF's entities in advance (Conf.)
• Researcher starts Slice creation procedures by delivering user's request (Rspec) to target CF

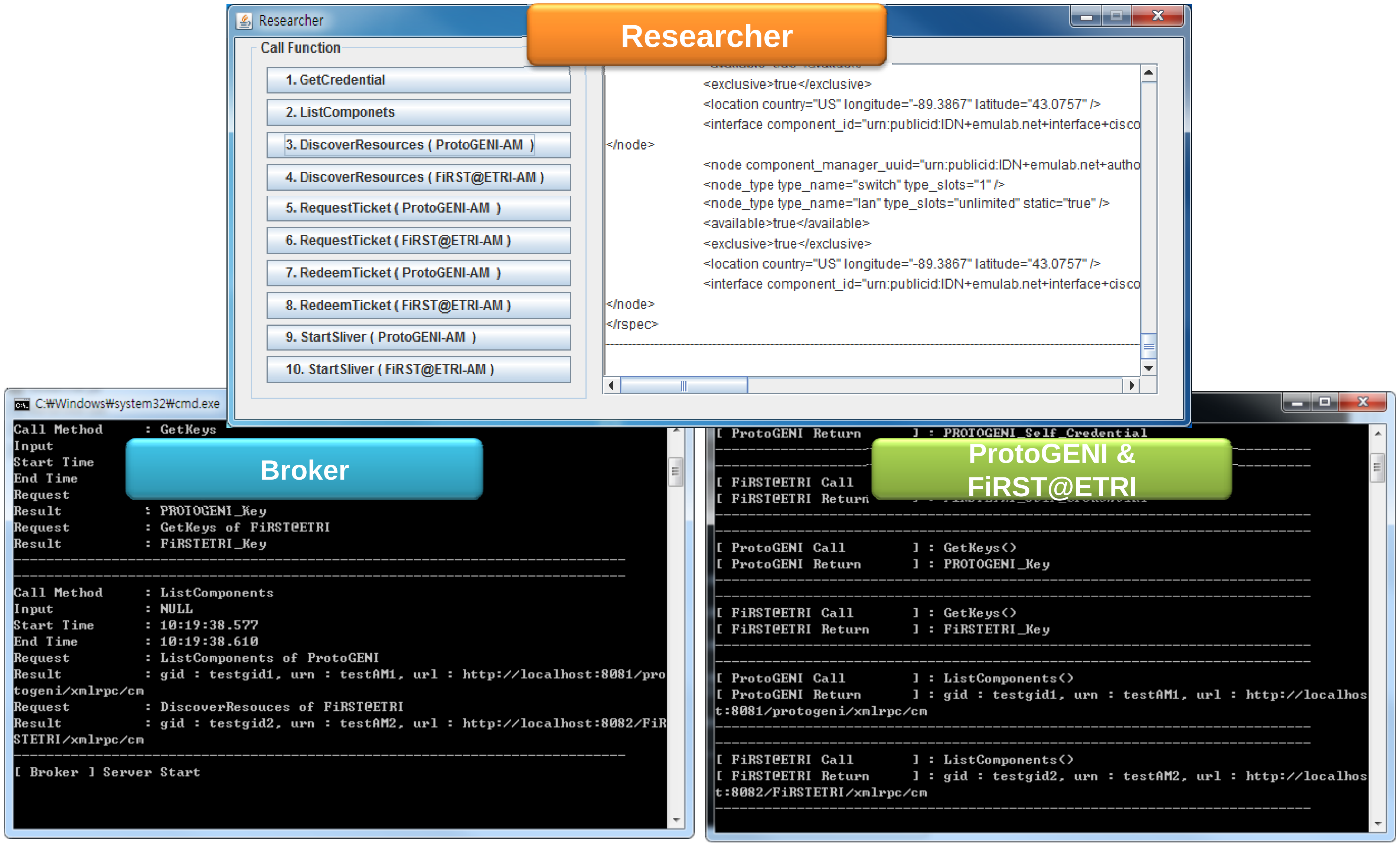
Functional architecture of broker



Issues for federation broker

- Broker performs API and RSpec translation
  - Researcher can use single UI for accessing different CFs or testbeds
  - Broker provides researchers with transparent access to resources in different CFs
- How to cope with heterogeneous federated requests?
  - The translation support for CFs should be flexible & manageable
- How to handle users credential?
- How to inject or negotiate access policy?

Feasibility test of broker


Researcher
Call Function
1. GetCredential
2. ListComponets
3. DiscoverResources (ProtoGENI-AM)
4. DiscoverResources (FiRST@ETRI-AM)
5. RequestTicket (ProtoGENI-AM)
6. RequestTicket (FiRST@ETRI-AM)
7. RedeemTicket (ProtoGENI-AM)
8. RedeemTicket (FiRST@ETRI-AM)
9. StartSliver (ProtoGENI-AM)
10. StartSliver (FiRST@ETRI-AM)
Researcher
C:₩Windows₩system32₩cmd.exe
Broker
ProtoGENI &
FiRST@ETRI

```
                <exclusive>true</exclusive>
                <location country="US" longitude="-89.3867" latitude="43.0757" />
                <interface component_id="urn:publicid:IDN+emulab.net+interface+cisco
</node>
                <node component_manager_uuid="urn:publicid:IDN+emulab.net+autho
                <node_type type_name="switch" type_slots="1" />
                <node_type type_name="lan" type_slots="unlimited" static="true" />
                <available>true</available>
                <exclusive>true</exclusive>
                <location country="US" longitude="-89.3867" latitude="43.0757" />
                <interface component_id="urn:publicid:IDN+emulab.net+interface+cisco
</node>
</rspec>
-------------------------------------------------------------------------------
```

```
Call Method       : GetKeys
Input
Start Time
End Time
Request
Result            : PROTOGENI_Key
Request           : GetKeys of FiRST@ETRI
Result            : FiRSTETRI_Key
---------------------------------------------------------------------------
---------------------------------------------------------------------------
Call Method       : ListComponents
Input             : NULL
Start Time        : 10:19:38.577
End Time          : 10:19:38.610
Request           : ListComponents of ProtoGENI
Result            : gid : testgid1, urn : testAM1, url : http://localhost:8081/pro
togeni/xmlrpc/cm
Request           : DiscoverResouces of FiRST@ETRI
Result            : gid : testgid2, urn : testAM2, url : http://localhost:8082/FiR
STETRI/xmlrpc/cm
---------------------------------------------------------------------------
[ Broker ] Server Start
```

```
[ ProtoGENI  Return       ] : PROTOGENI_Self_Credential
-------------------------------------------------------------------------
-------------------------------------------------------------------------
[ FiRST@ETRI  Call
[ FiRST@ETRI  Return
-------------------------------------------------------------------------
-------------------------------------------------------------------------
[ ProtoGENI  Call         ] : GetKeys()
[ ProtoGENI  Return       ] : PROTOGENI_Key
-------------------------------------------------------------------------
-------------------------------------------------------------------------
[ FiRST@ETRI  Call        ] : GetKeys()
[ FiRST@ETRI  Return      ] : FiRSTETRI_Key
-------------------------------------------------------------------------
-------------------------------------------------------------------------
[ ProtoGENI  Call         ] : ListComponents()
[ ProtoGENI  Return       ] : gid : testgid1, urn : testAM1, url : http://localhos
t:8081/protogeni/xmlrpc/cm
-------------------------------------------------------------------------
-------------------------------------------------------------------------
[ FiRST@ETRI  Call        ] : ListComponents()
[ FiRST@ETRI  Return      ] : gid : testgid2, urn : testAM2, url : http://localhos
t:8082/FiRSTETRI/xmlrpc/cm
-------------------------------------------------------------------------
```

Way forward

- Open issues
  - Credential issues

- Next steps
  - Finalize broker architecture
  - Implement prototype system (in Java)