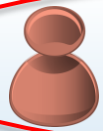
Two Roles with "Administration" Responsibility

- Network Service Provider
  - **Governs** the network: channels, membership etc.
  - A consortium of network members or designated authority

- Network Service Consumer
  - **Operates** a set of peers and certificate authorities on the network
  - Represents an organization on the business network

- Business Service Provider
  - **Develops** blockchain business applications
  - Includes transaction, app server, integration and presentation logic

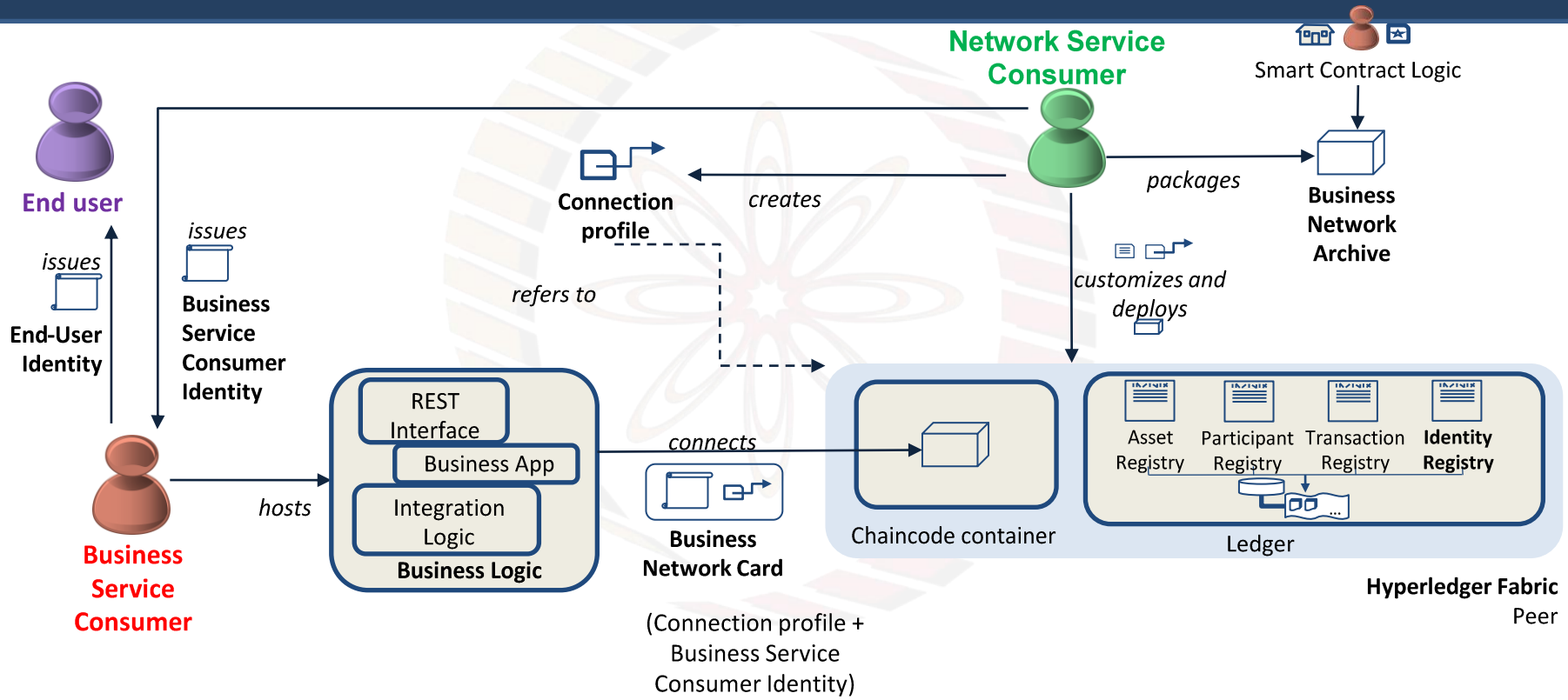


- Business Service Consumer
  - Hosts application and integration logic which invokes blockchain transactions

- End-user
  - Runs presentation logic e.g. on mobile device or dashboard

A single organization may play multiple roles!

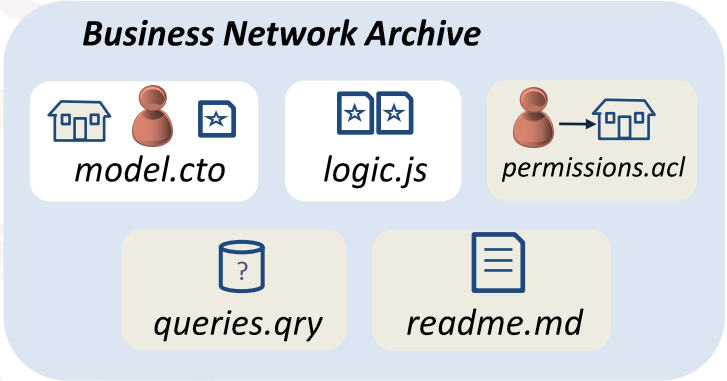
Key Concepts for Administrators



Network Service Consumer packages resources in a BNA file

- Business Network Archive (.BNA) is a package of the resources used by Fabric:
  - Model files (.CTO)
  - Transaction processors (.JS)
  - Access Control Lists (.ACL)
  - Static queries (.QRY)
  - Documentation and versioning (.MD)
  - It does *not* contain the client application

model files contains description of participants and assets

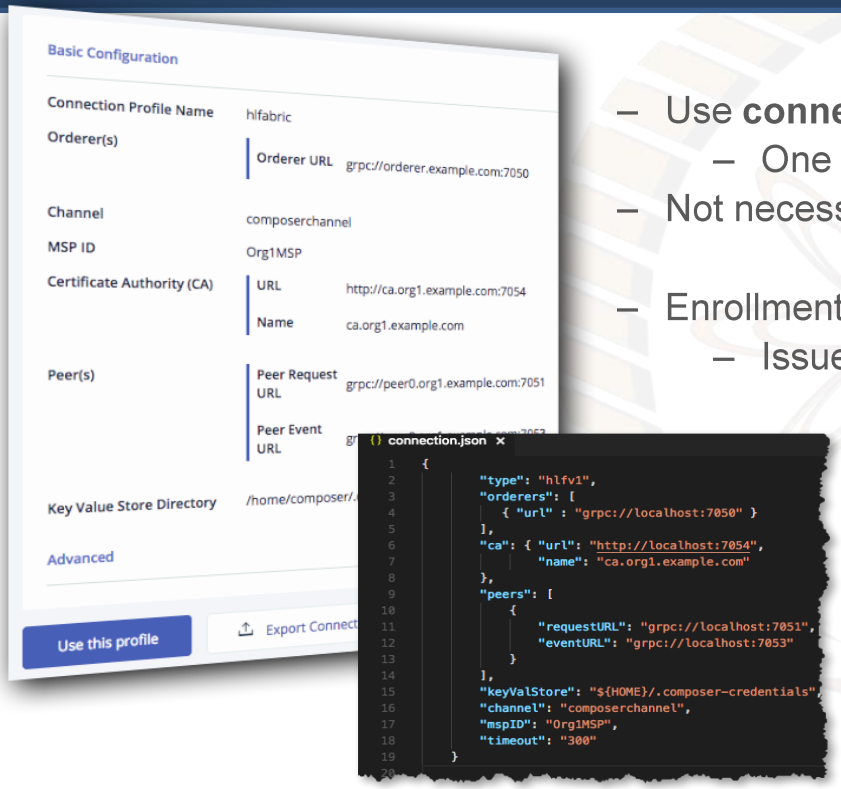


- The BNA simplifies deployment of blockchain and promotion between environments
  - c.f. TAR, WAR, EAR, JAR, BAR…

```
composer archive create –archiveFile my.bna
                --sourceType module --sourceName myNetwork
```

- Create BNA files from Playground or command line
  - Build from filesystem or NPM module

Connection Profiles to Hyperledger Fabric



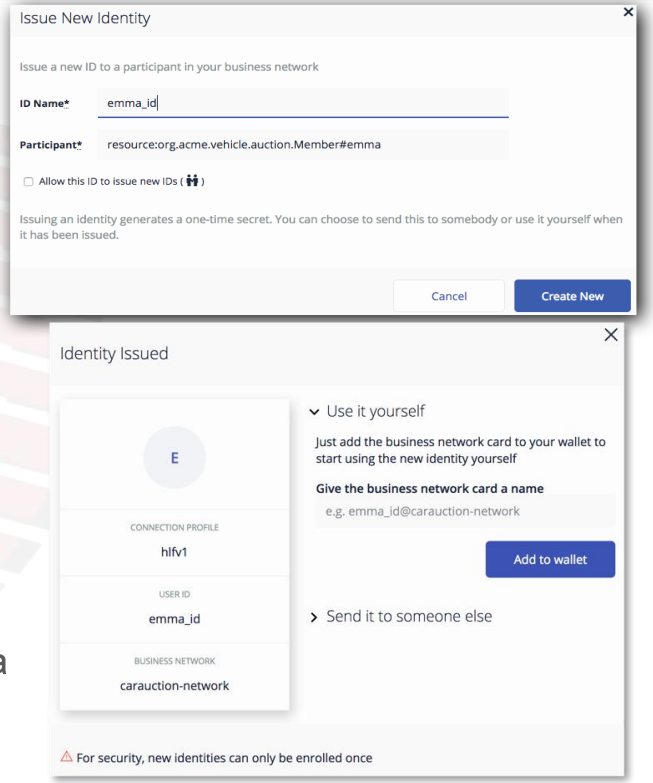
- Use **connection profiles** to describe Fabric connection parameters
  - One connection profile required per channel
- Not necessary for web-based simulation

- Enrollment in Hyperledger Fabric network required (see later)
  - Issue Fabric identity from Composer participants

- Connection profiles currently used by Composer only
  - Plans to implement common connection profiles that can be used by both Fabric and Composer

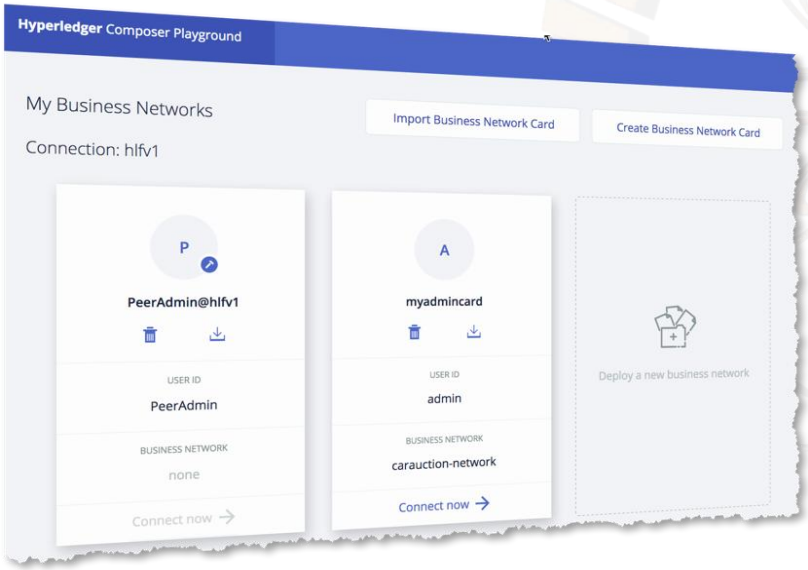
Participant Identity

- The Network Service Consumer issues network participants with an **identity** in order to connect to Hyperledger Fabric
  - Issued as a Hyperledger Fabric userid/secret
  - Automatically swapped for a certificate on first use
  - Packaged in a Business Network Card and supplied when the client application connects
- Composer Participant to Fabric Identity mapping is stored on the blockchain in an *identity registry*
- Usually, only Business Service Consumers have a Fabric identity
  - End-users log in to the business application using a separately managed identity; blockchain transactions invoked by proxy
- Manage identity from Playground, Javascript, REST or command line
  - For example: Test connection, issue identity, bind an identity to a participant, revoke an identity, list identities



Business Network Cards

- Business Network Cards are a convenient packaging of *identity* and *connection profile*
  - Contains everything you need to connect to blockchain business network
  - Each card refers to a single participant and single business network
  - Analogous to an ATM card



- Manage cards from both Playground and command-line
  - Create, delete, export, import, list
  - Create requires userid/secret or certificate/private key
- Use cards to connect to Fabric from Playground, command-line or from within your application

```
// Connect and log in to HLF
var businessNetwork = new BusinessNetworkConnection();
return businessNetwork.connect('cardName')
.then(function(businessNetworkDefinition){
    // Connected
});
```

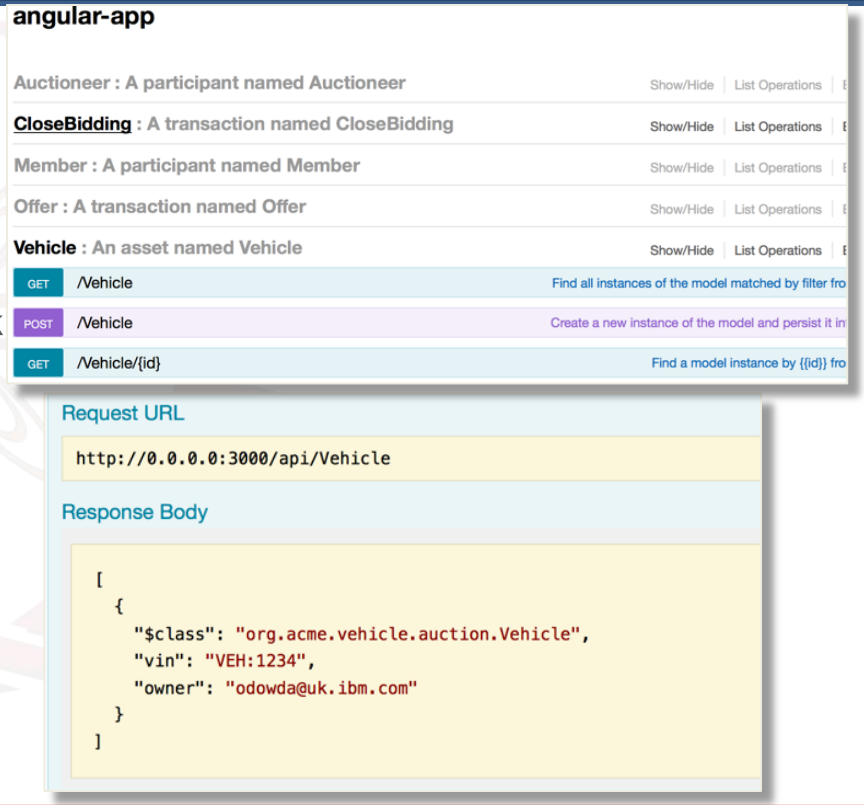
Systems of Record Integration

- Domain specific APIs very attractive to mobile and web developers. Resources and operations are business-meaningful
- Composer exploits Loopback framework to create REST APIs: https://loopback.io/

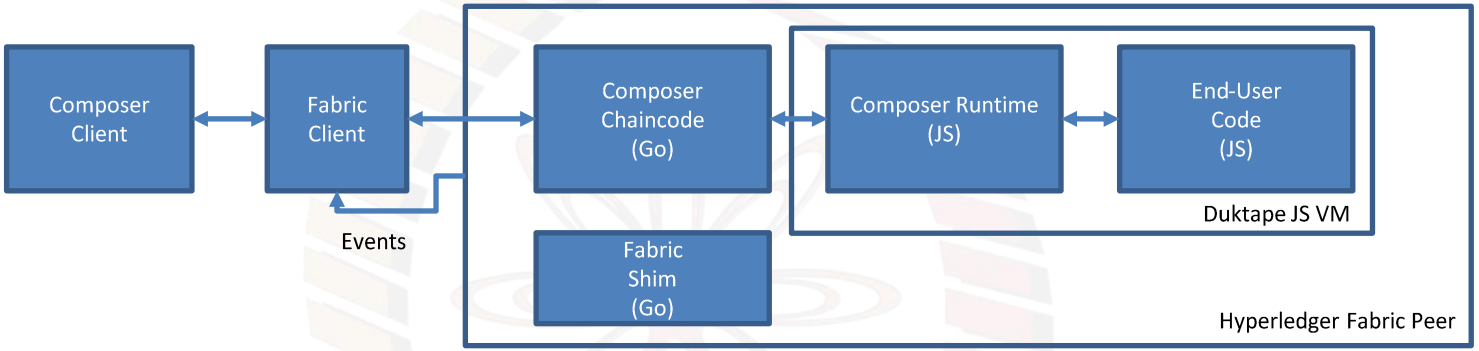
- Extensive test facilities for REST methods using loopback

- Secured using JS Passport, giving >400 options for authentication

- Composer provides back-end integration with any loopback compatible product
  - e.g. IBM Integration Bus, API Connect, StrongLoop
  - Outbound and Inbound (where supported by middleware)



How Composer Maps to Fabric Chaincode



- Each Business Network is deployed to its own chaincode container
  - Container contains a static piece of Go chaincode that starts a Javascript virtual machine running transaction processors
- Browse these containers to view diagnostic information (docker logs)
- Embedded chaincode is not a Composer external interface