chaincode management in 2.x

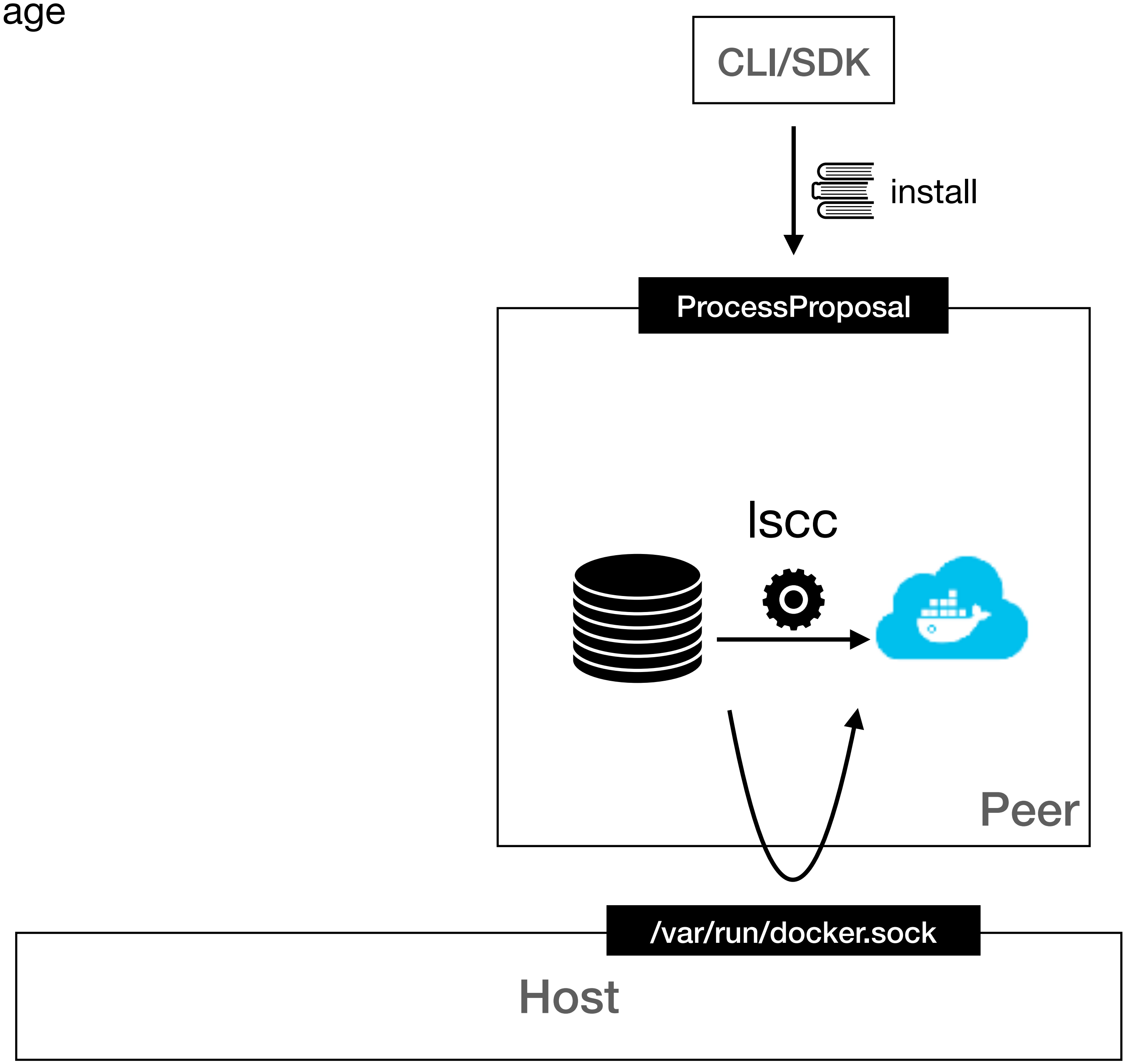
郭剑南 guojiannan@cn.ibm.com

Outline

- 1.x版本链码部署过程
- 存在的问题
- 2.0版本中增加的部署方式

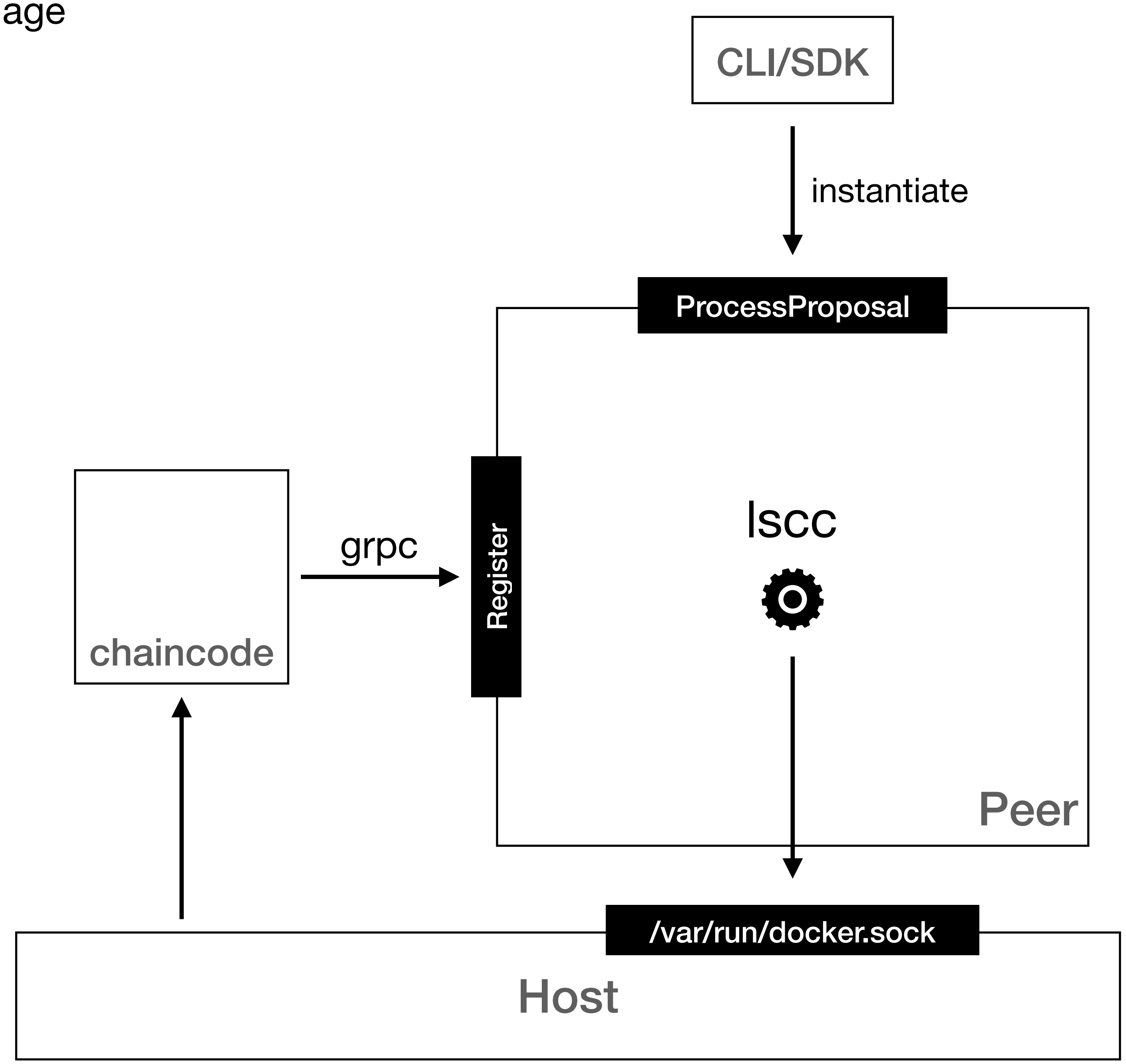
1.x版本中链码的部署过程

1. install pkg to peer
   - calls lscc via ProcessProposal
   - store pkg to filesystem of peer
   - build source code and produce docker image
2. instantiate chaincode
   - calls lscc via ProcessProposal
   - lookup img and launch container
   - wait for container to register with peer
   - communicate via established grpc conn
   - calls init if necessary



1.x版本中链码的部署过程

1. install pkg to peer
   - calls lscc via ProcessProposal
   - store pkg to filesystem of peer
   - build source code and produce docker image
2. instantiate chaincode
   - calls lscc via ProcessProposal
   - lookup img and launch container
   - wait for container to register with peer
   - communicate via established grpc conn
   - calls init if necessary



Problems

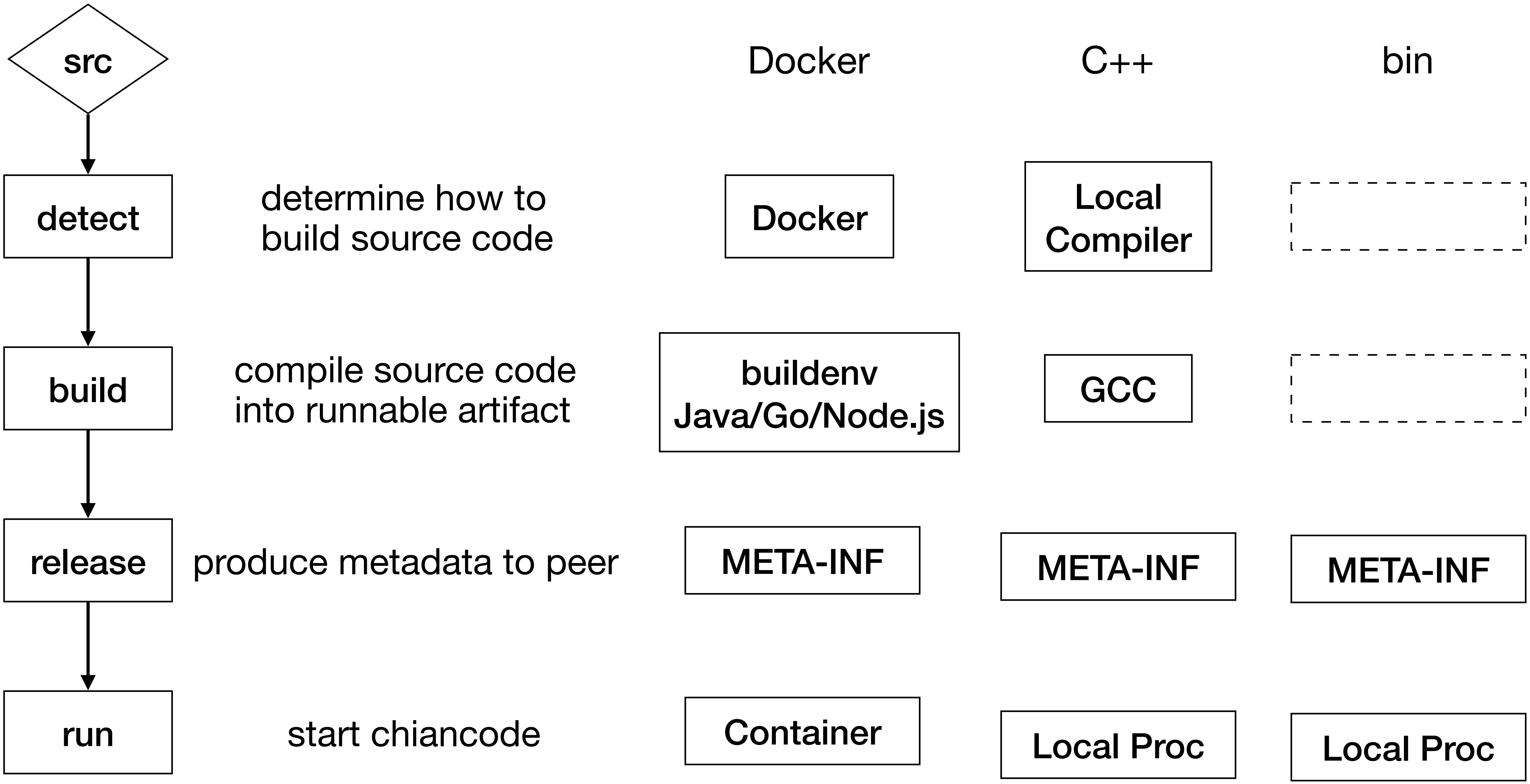
- complexity in managing chaincode container lifecycle
  - cannot provision or scale containers with standard cloudnative tools, i.e. k8s
- elevated privilege is required, not complying with common practice
  - peer has access to host docker daemon (sometimes mitigated by docker in docker, aka dind)

peers should be able to decide individually how to build and run chaincode

** idea loosely based on Heroku Buildpacks*

Buildpack

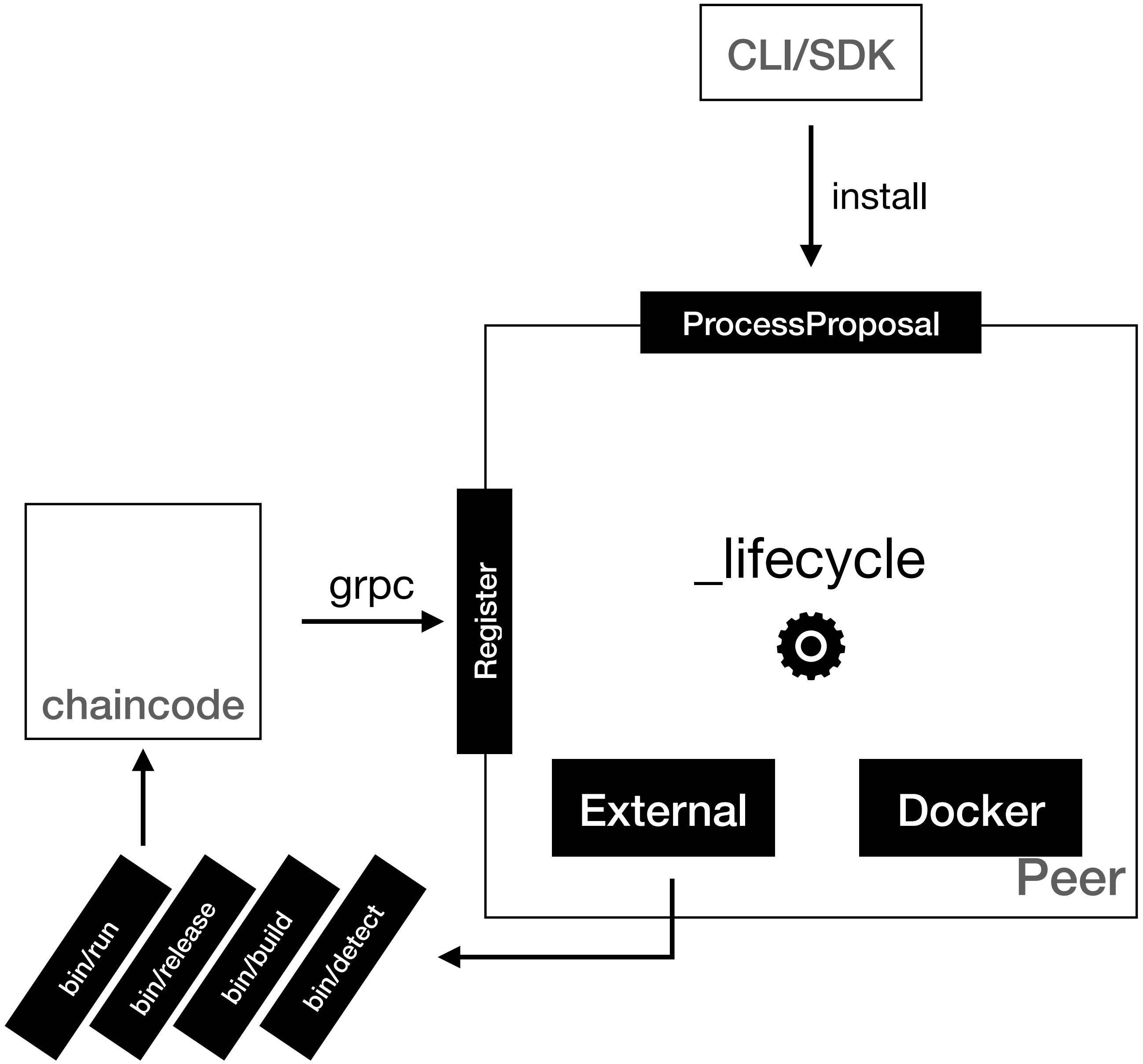
From source code to running process



src		Docker	C++	bin
detect	determine how to build source code	Docker	Local Compiler	
build	compile source code into runnable artifact	buildenv Java/Go/Node.js	GCC	
release	produce metadata to peer	META-INF	META-INF	META-INF
run	start chiancode	Container	Local Proc	Local Proc

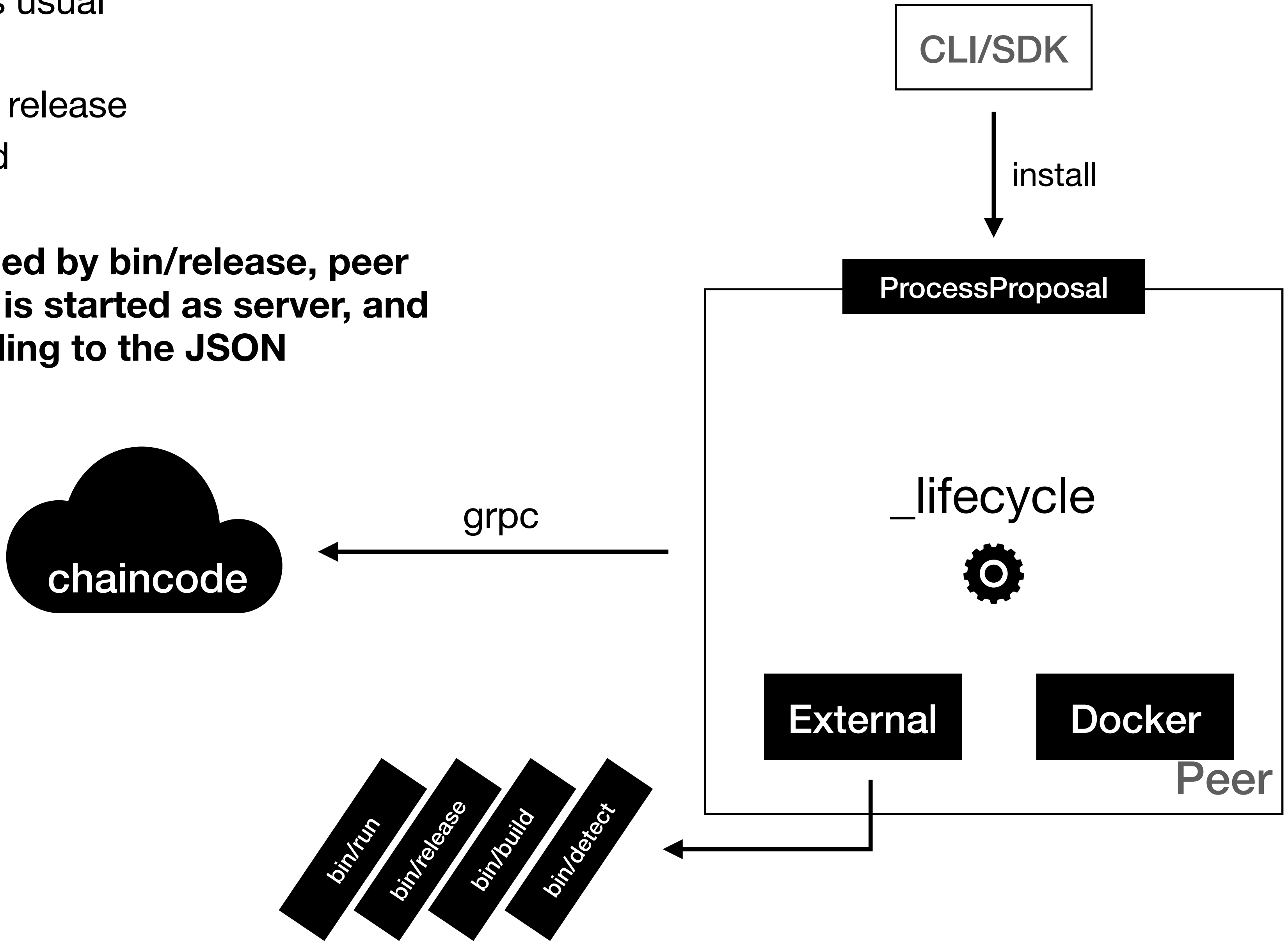
2.0版本中链码的部署过程

1. configure external builders for peer (simple scripts or complex binary)
2. install via ProcessProposal as usual
   - detect external builder
   - run through build process till release
3. start chaincode when invoked
   - start with bin/run



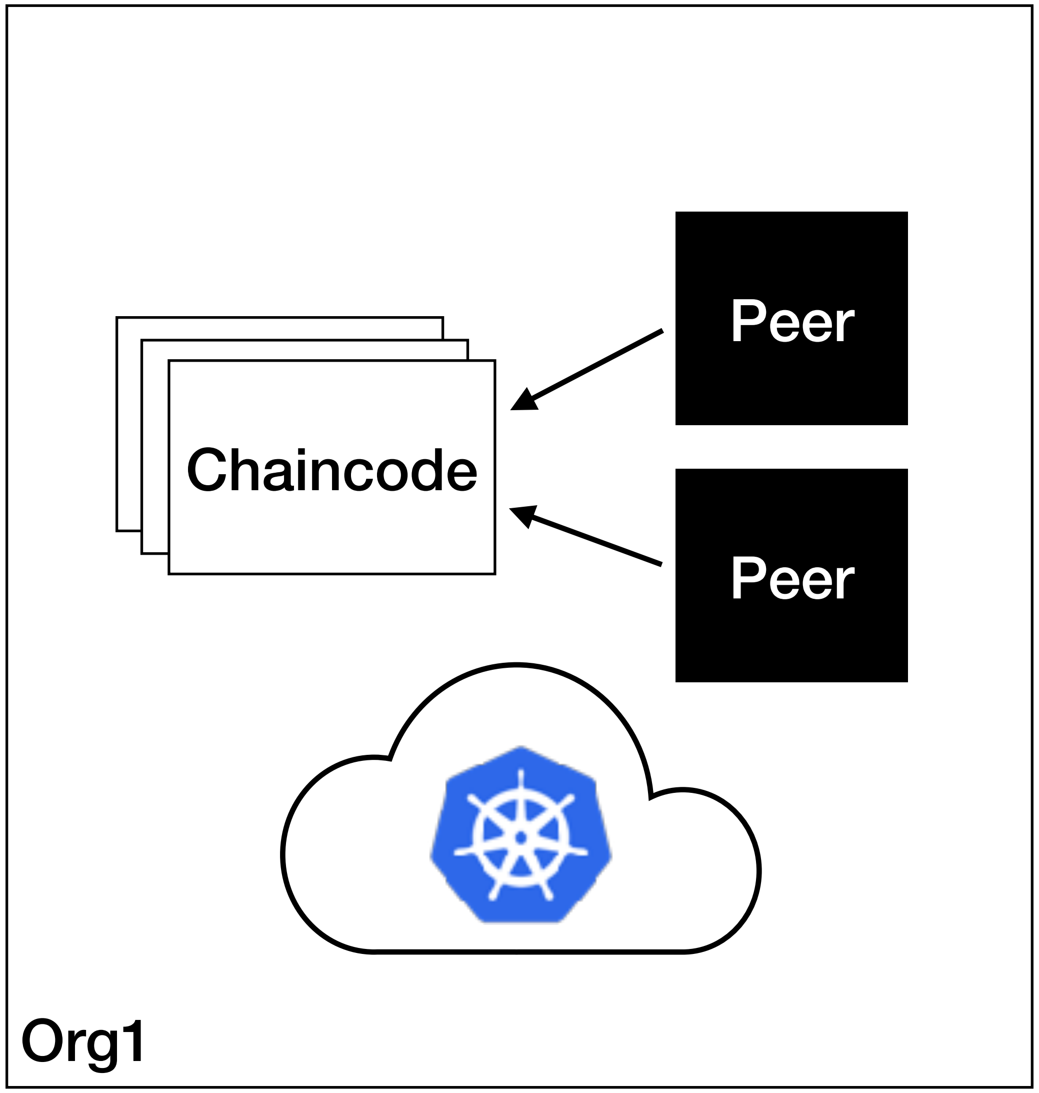
2.0版本中链码的部署过程

1. configure external builders for peer (simple scripts or complex binary)
2. install via ProcessProposal as usual
   - detect external builder
   - run through build process till release
3. start chaincode when invoked
   - start with bin/run
   - **if connection.json is provided by bin/release, peer recognizes that chaincode is started as server, and initiates connection according to the JSON**



* example of running smart contract on knative (k8s based serverless framework): https://www.youtube.com/watch?v=pMuJQutqt7c

Big picture


Peer
Chaincode
Peer
Org1