SignArch

An architectural description of Signal Android

Duan

Maximilian

Maksim

Mathieu

Dong

Signal: A messaging app known for privacy

Fast, simple, secure.

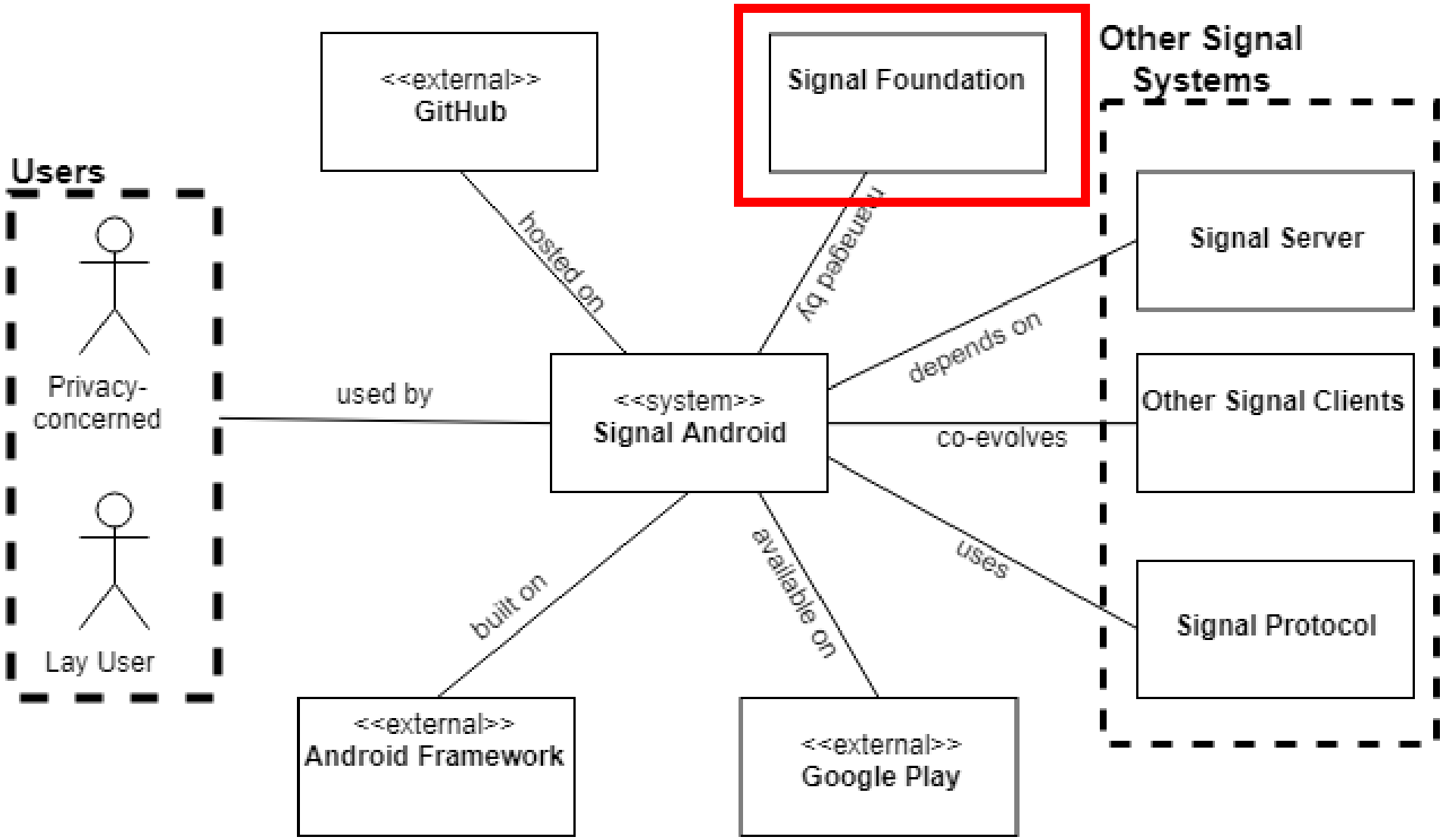
Privacy that fits in your pocket.

Android

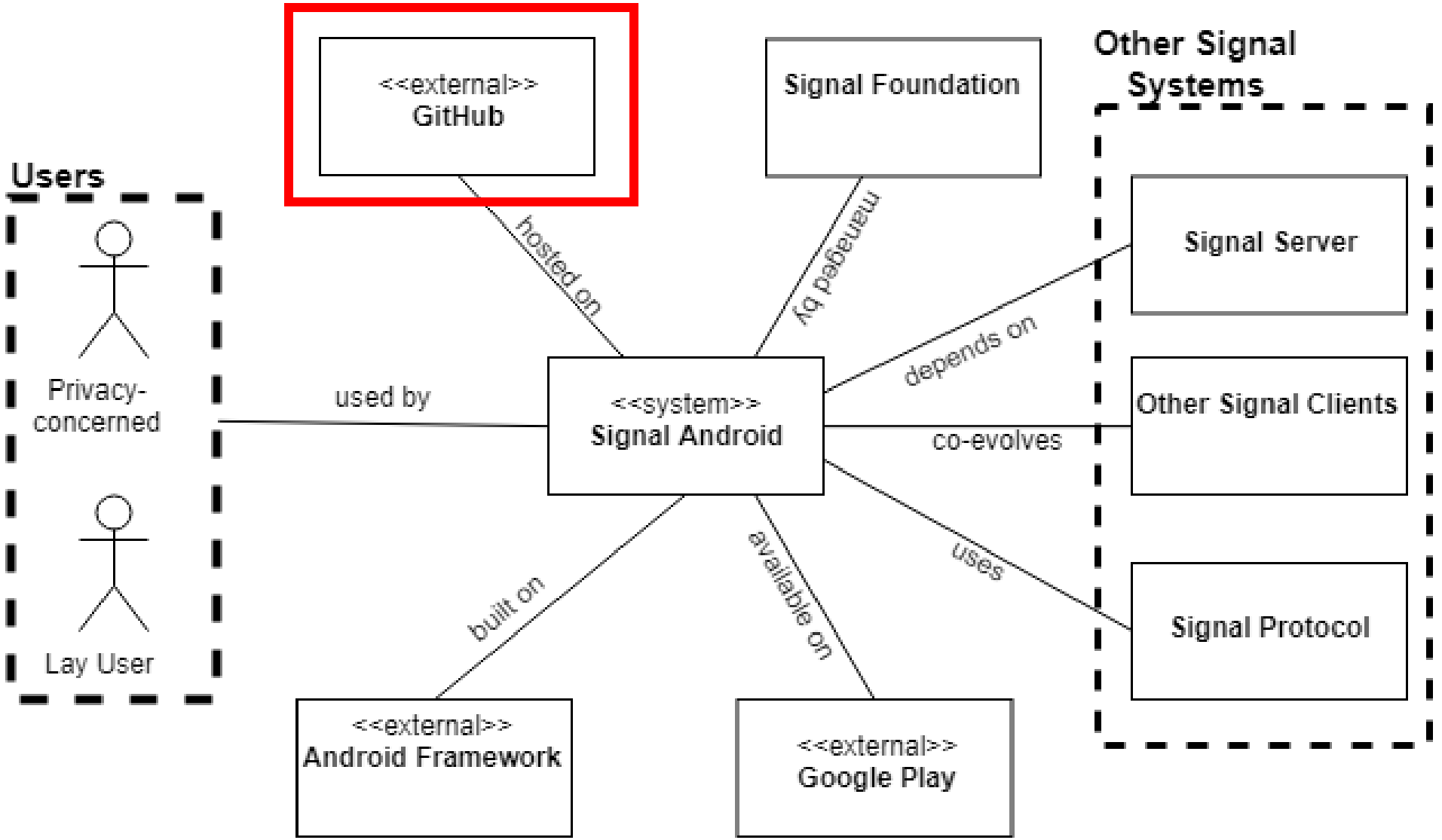
iPhone

Desktop

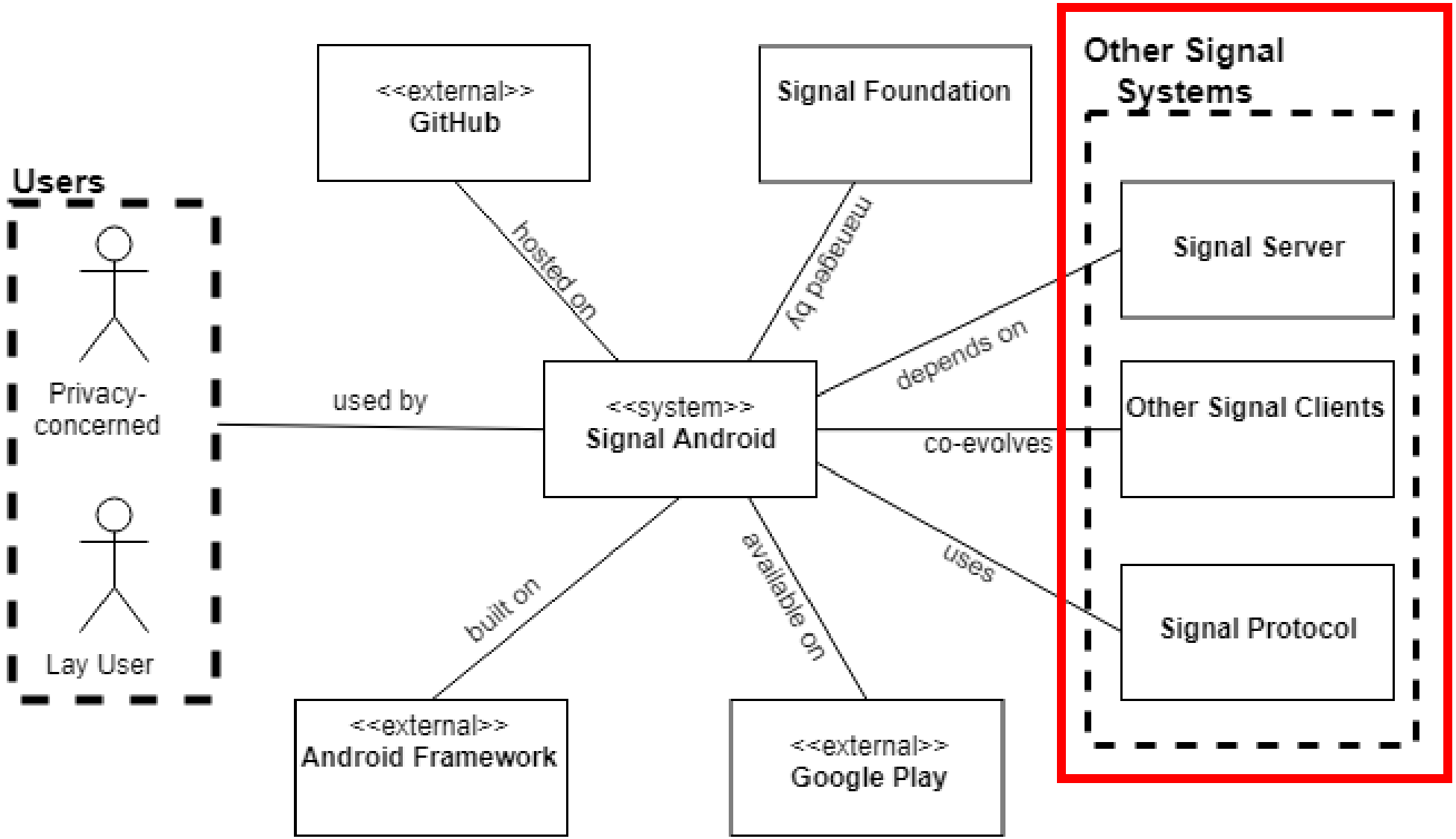
Signal Android



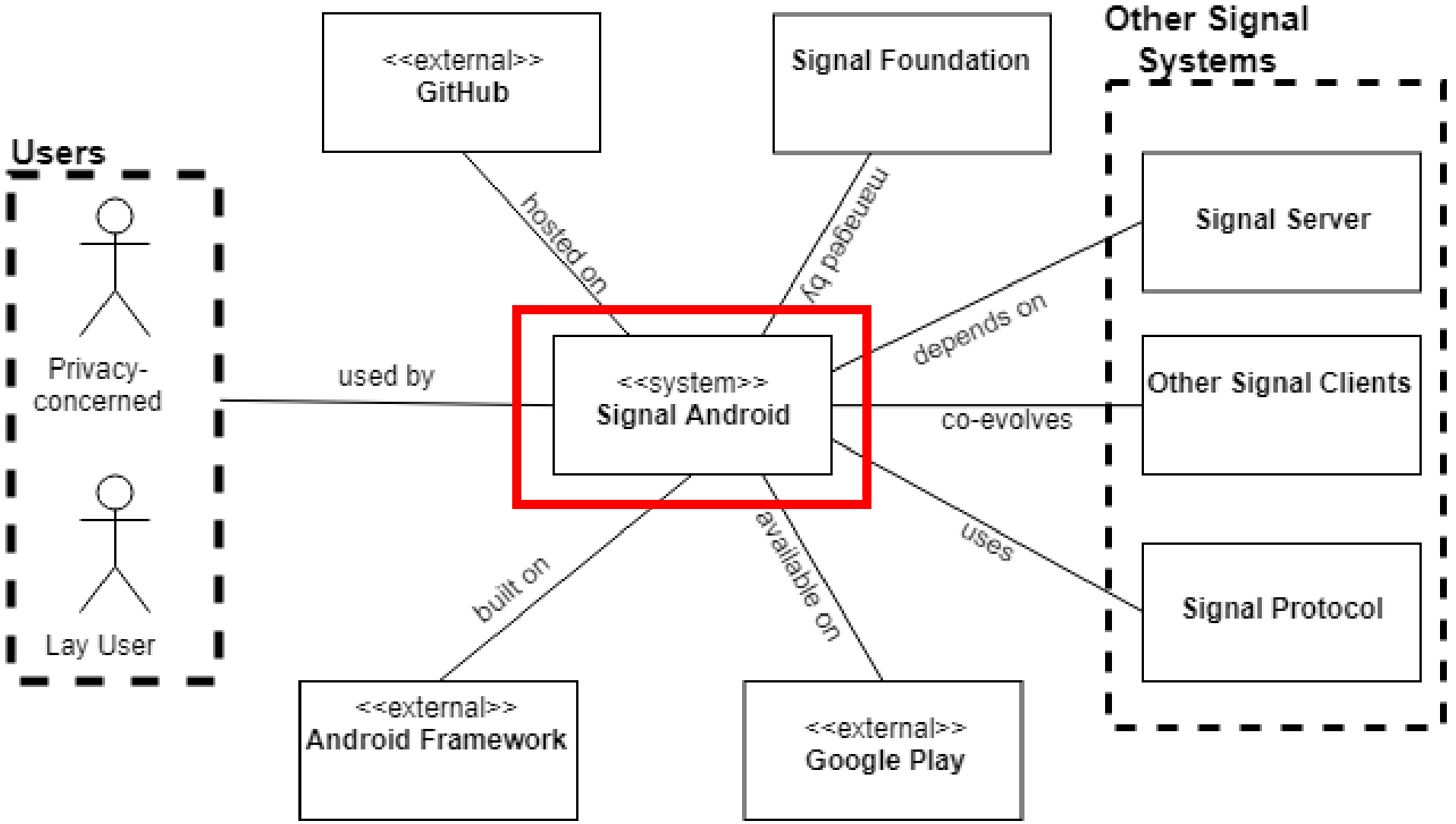
Signal Android



Signal Android



Signal Android



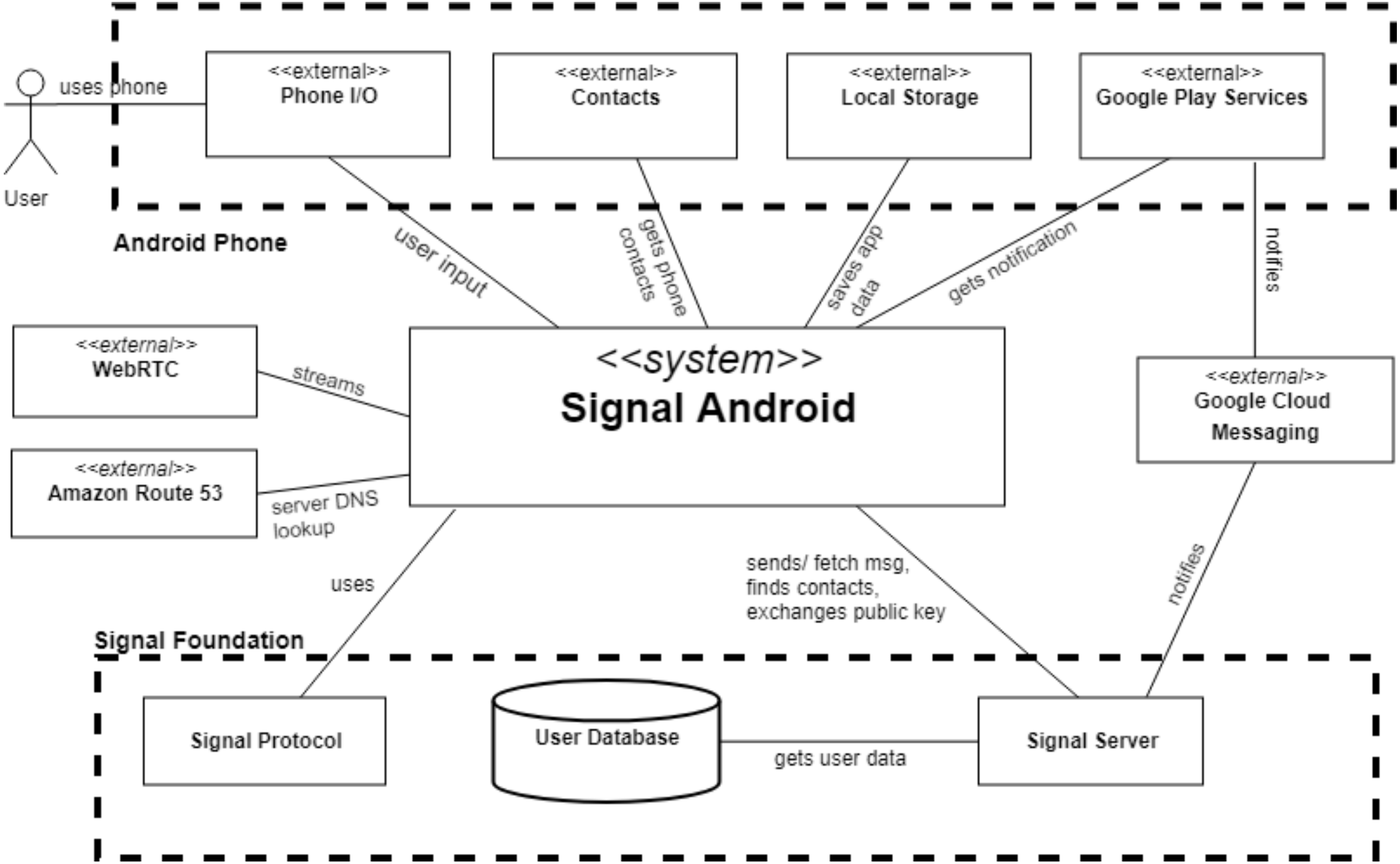
Requirements

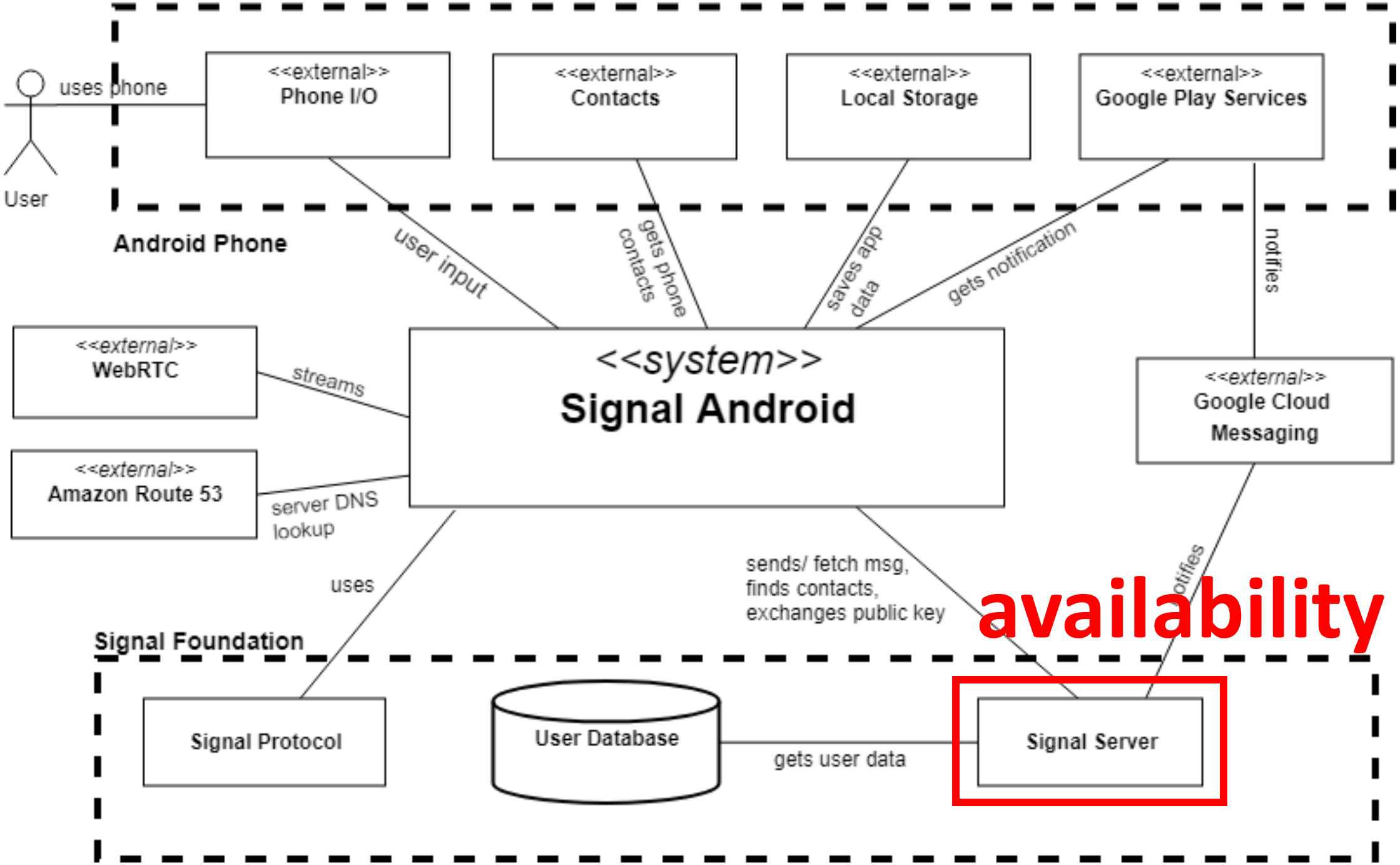
Functional Requirements:

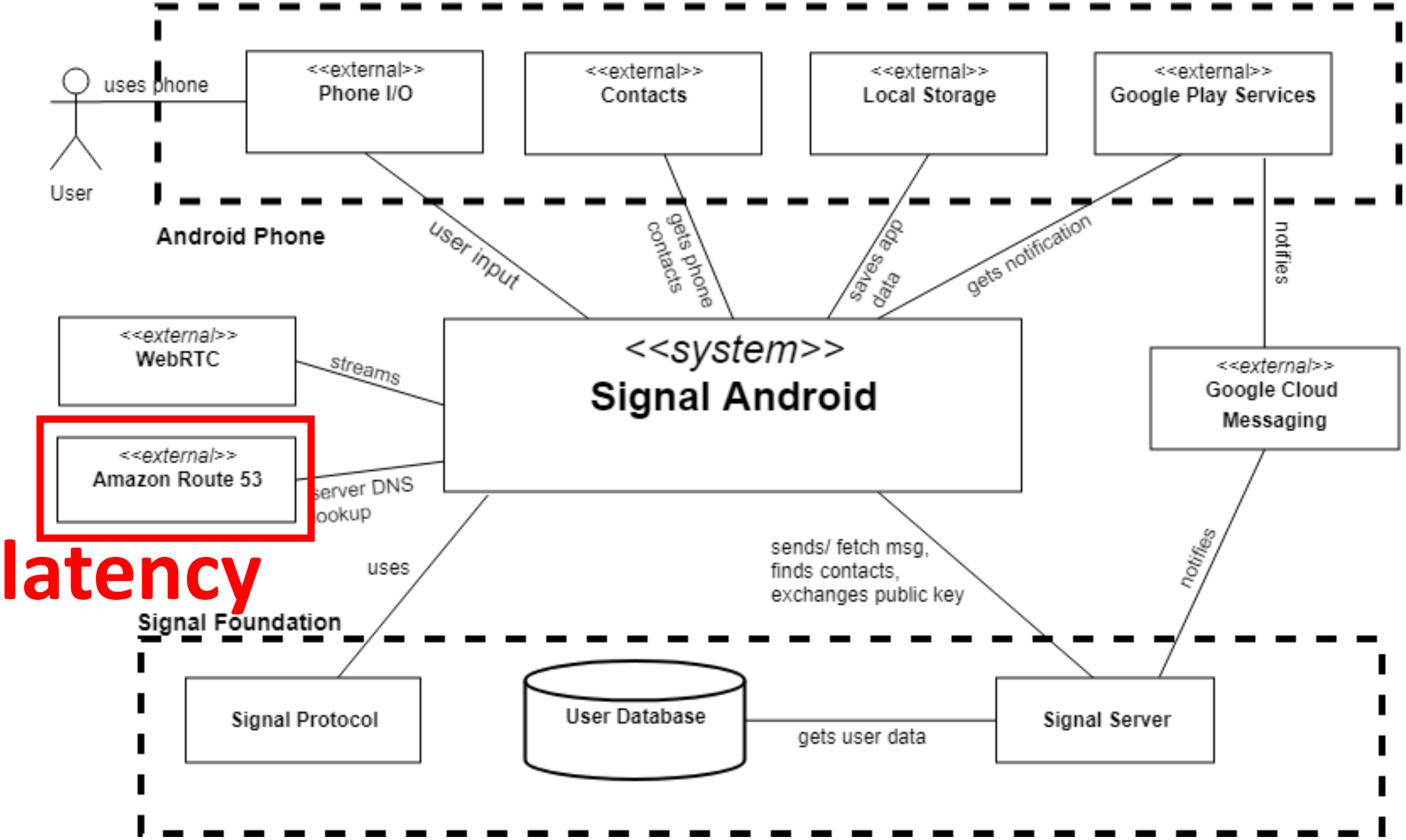
- Text messaging
- Voice chat
- Video chat

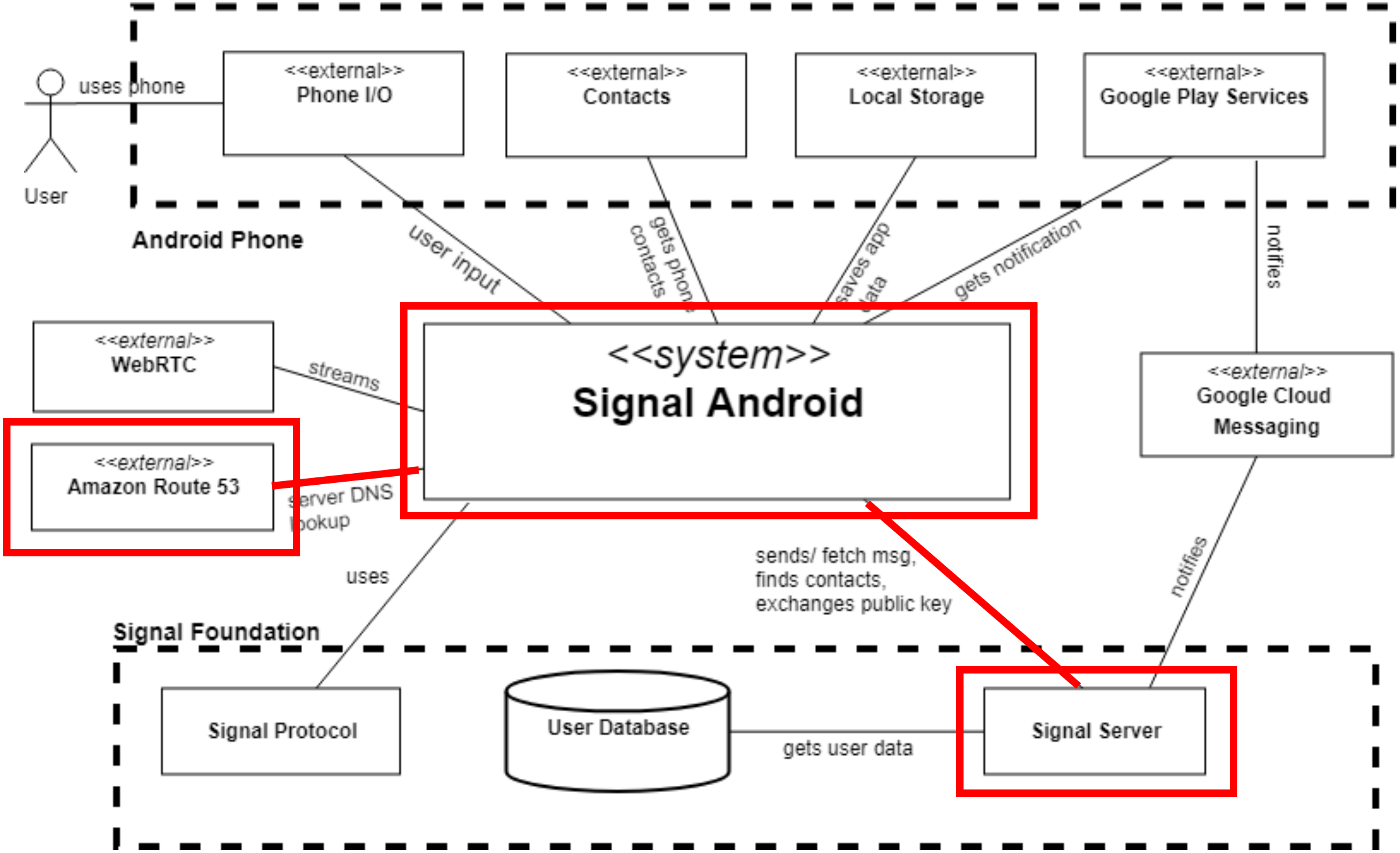
System qualities:

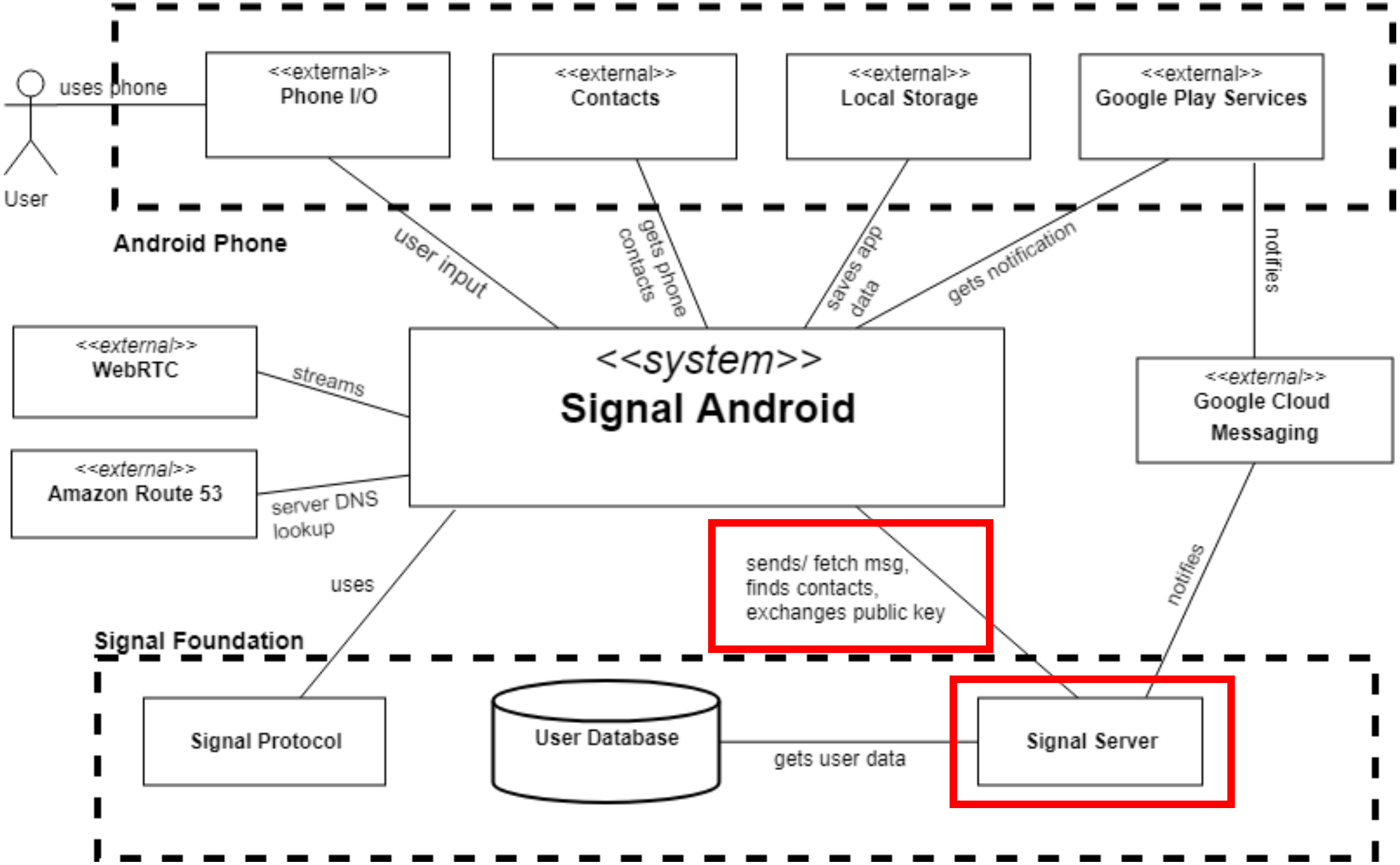
- Private, secure
- High availability
- High-quality, low-latency
- Evolvability

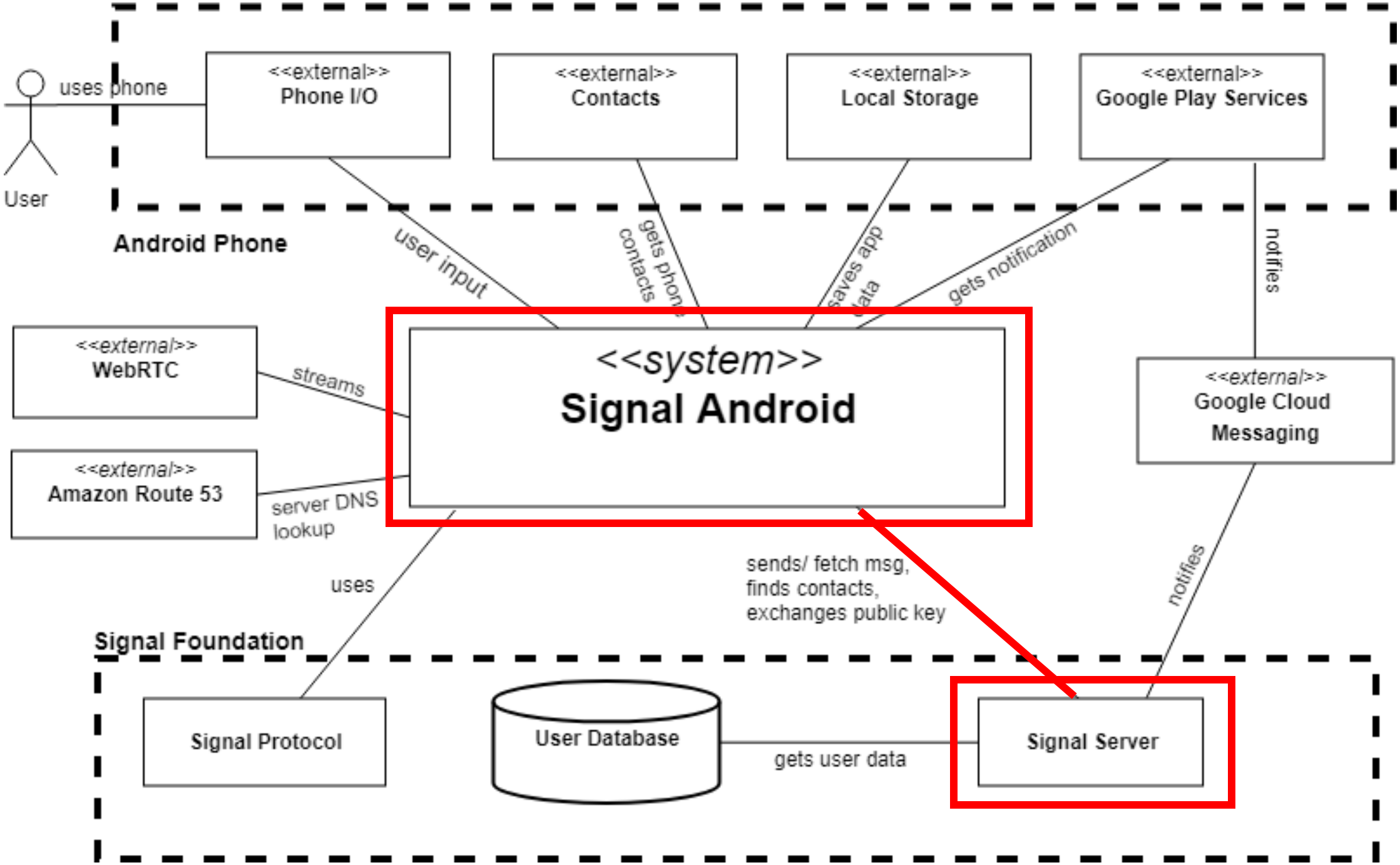

uses phone
User
<<external>>
Phone I/O
<<external>>
Contacts
<<external>>
Local Storage
<<external>>
Google Play Services
Android Phone
user input
gets phone contacts
saves app data
gets notification
notifies
<<external>>
WebRTC
streams
<<system>>
Signal Android
<<external>>
Google Cloud Messaging
<<external>>
Amazon Route 53
server DNS lookup
uses
sends/ fetch msg,
finds contacts,
exchanges public key
notifies
Signal Foundation
Signal Protocol
User Database
gets user data
Signal Server

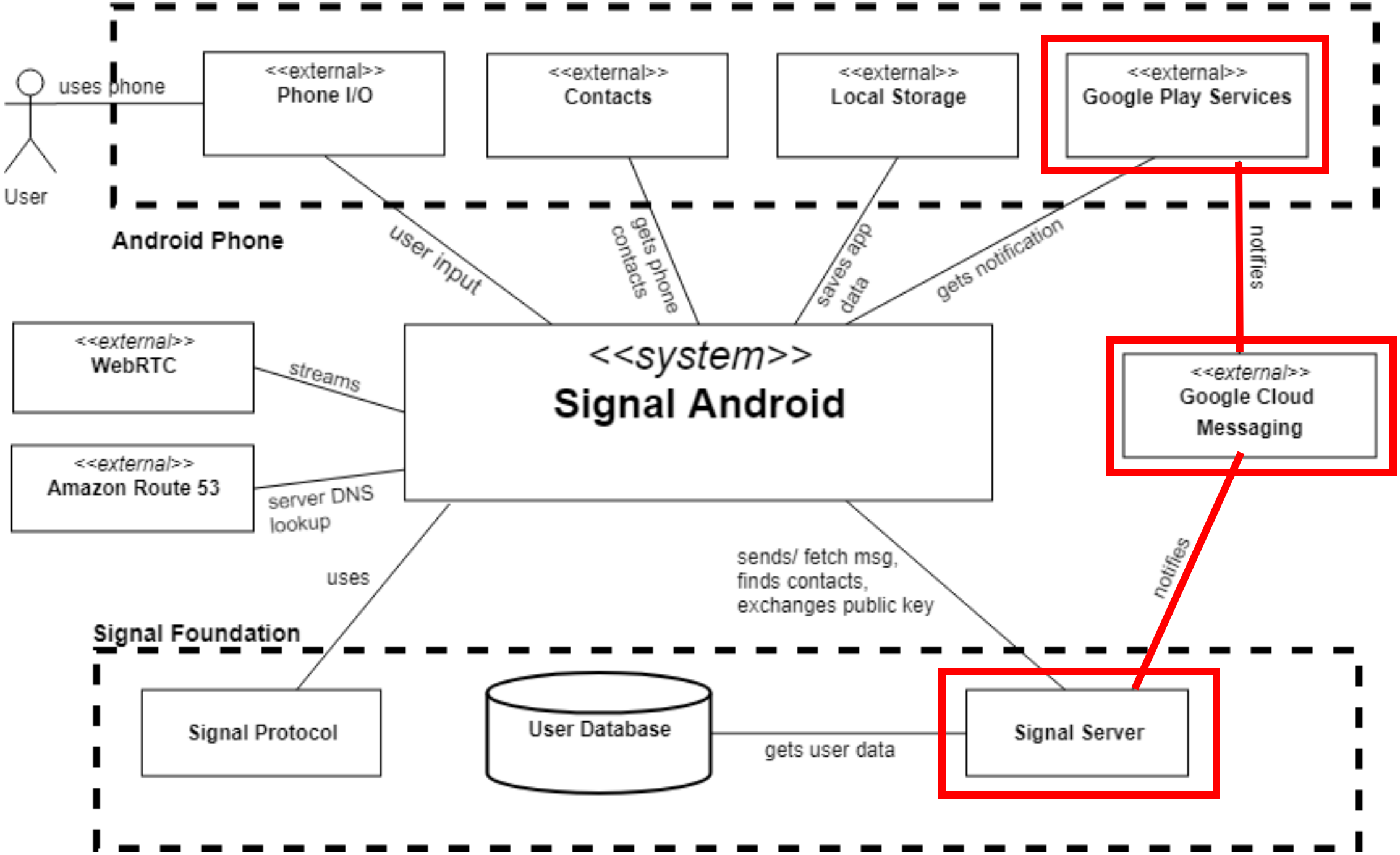

uses phone
User
<<external>>
Phone I/O
<<external>>
Contacts
<<external>>
Local Storage
<<external>>
Google Play Services
Android Phone
user input
gets phone contacts
saves app data
gets notification
notifies
<<external>>
WebRTC
streams
<<system>>
Signal Android
<<external>>
Google Cloud Messaging
<<external>>
Amazon Route 53
server DNS lookup
uses
sends/ fetch msg,
finds contacts,
exchanges public key
notifies
availability
Signal Foundation
Signal Protocol
User Database
gets user data
Signal Server

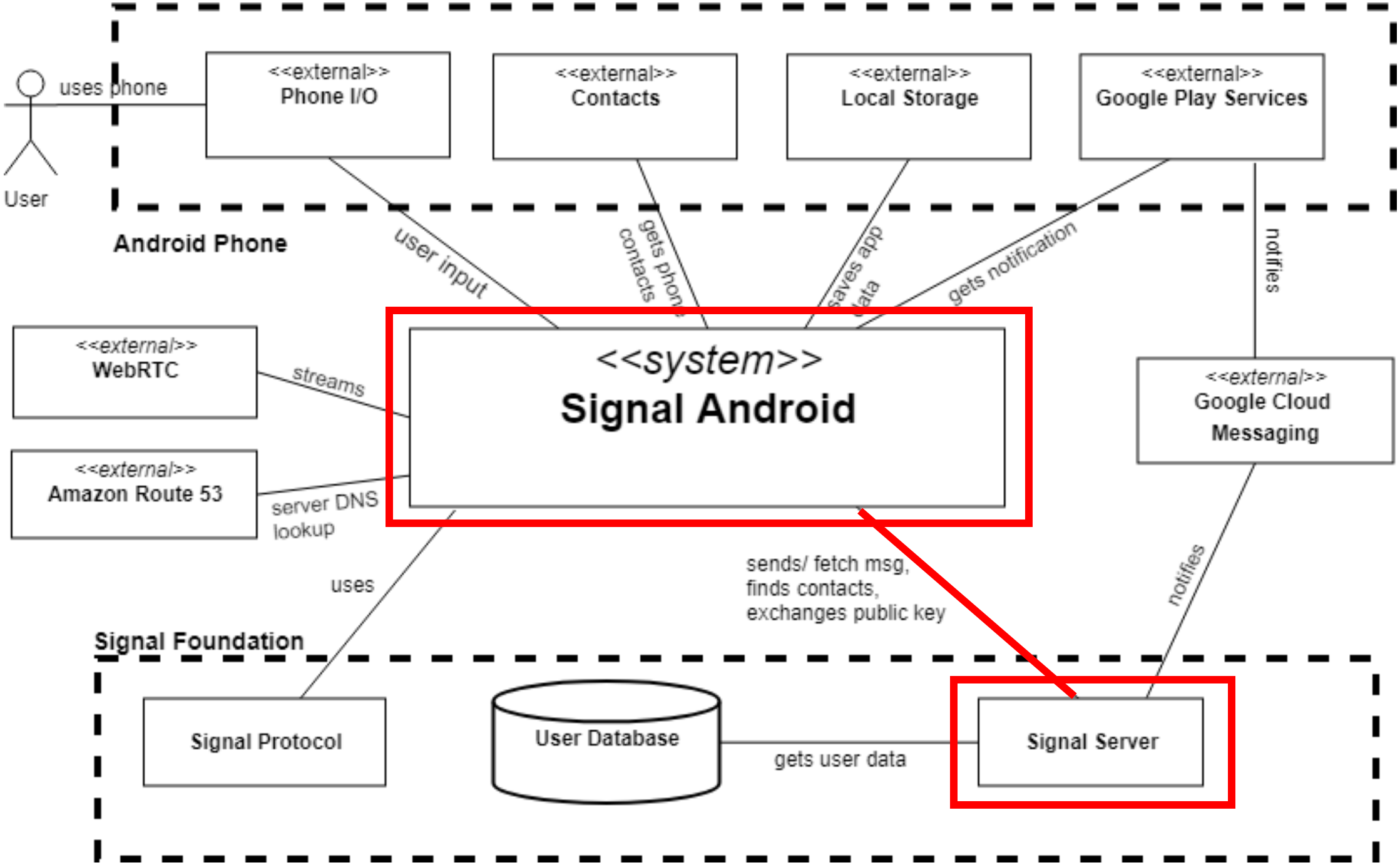

uses phone
User
<<external>>
Phone I/O
<<external>>
Contacts
<<external>>
Local Storage
<<external>>
Google Play Services
Android Phone
user input
gets phone contacts
saves app data
gets notification
notifies
<<external>>
WebRTC
streams
<<system>>
Signal Android
<<external>>
Google Cloud Messaging
<<external>>
Amazon Route 53
server DNS lookup
latency
uses
sends/ fetch msg,
finds contacts,
exchanges public key
notifies
Signal Foundation
Signal Protocol
User Database
gets user data
Signal Server

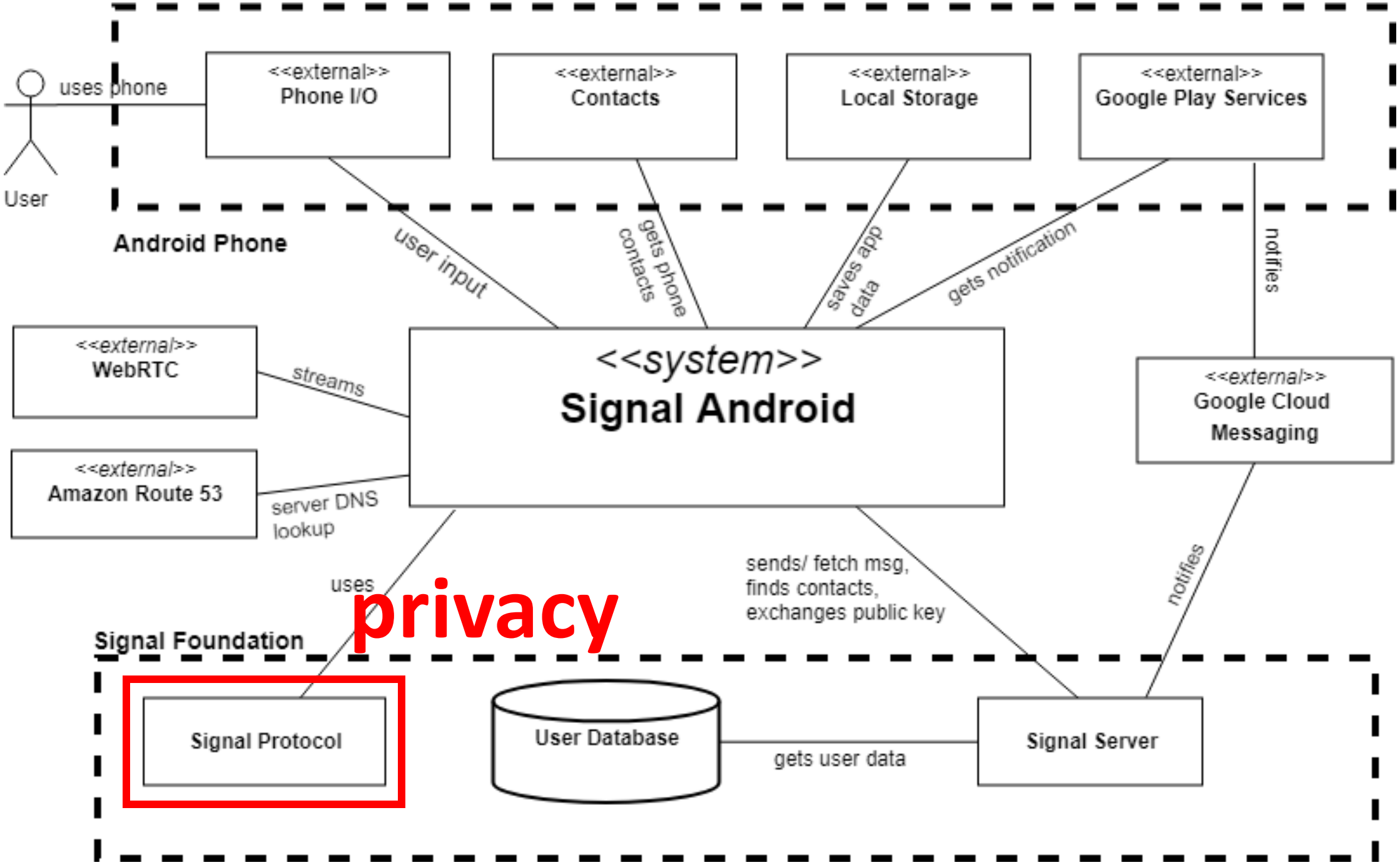

uses phone
User
<<external>>
Phone I/O
<<external>>
Contacts
<<external>>
Local Storage
<<external>>
Google Play Services
Android Phone
user input
gets phone contacts
saves app data
gets notification
notifies
<<external>>
WebRTC
streams
<<system>>
Signal Android
<<external>>
Google Cloud Messaging
<<external>>
Amazon Route 53
server DNS lookup
uses
sends/ fetch msg,
finds contacts,
exchanges public key
notifies
Signal Foundation
Signal Protocol
User Database
gets user data
Signal Server

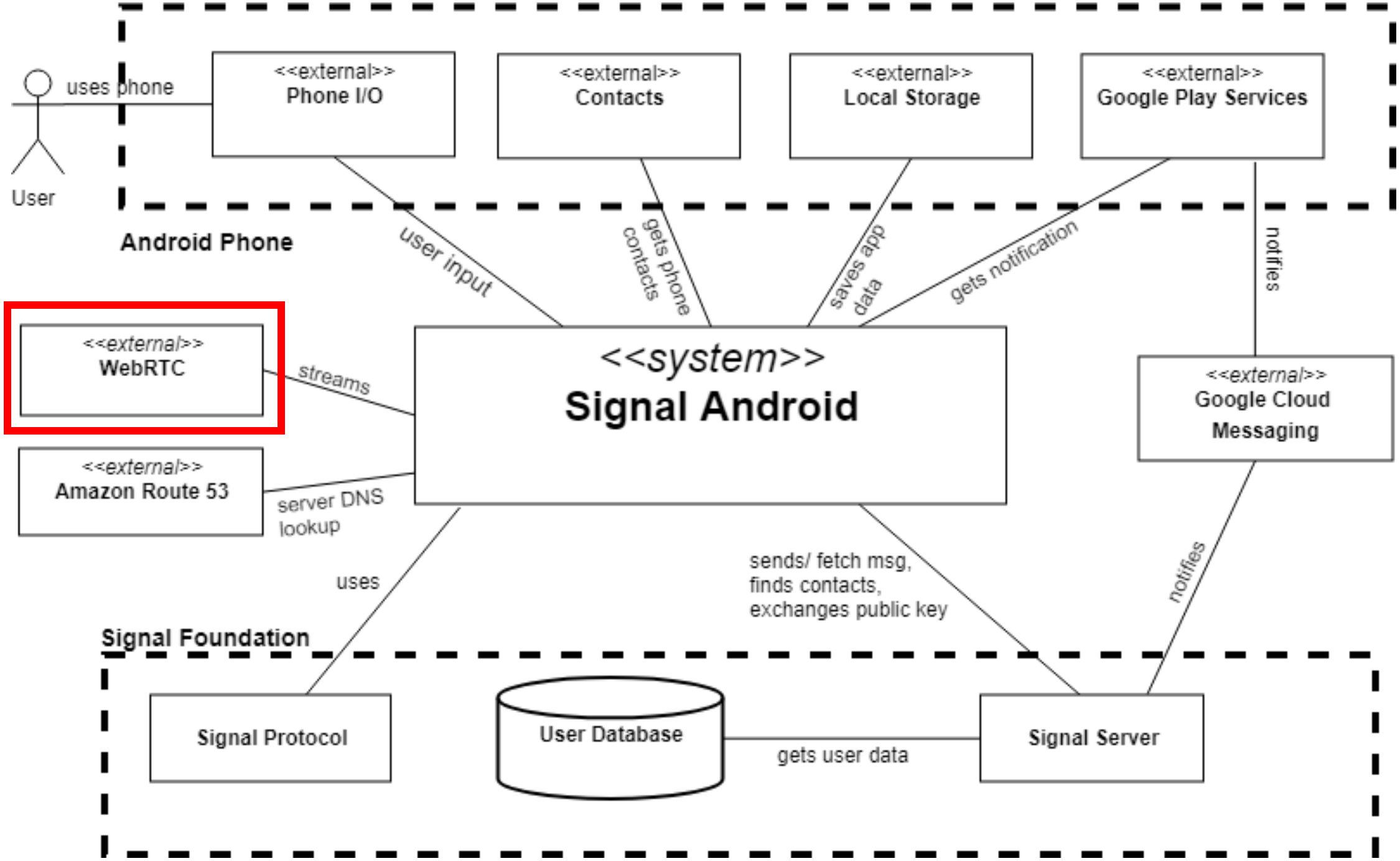

uses phone
User
<<external>>
Phone I/O
<<external>>
Contacts
<<external>>
Local Storage
<<external>>
Google Play Services
Android Phone
user input
gets phone contacts
saves app data
gets notification
notifies
<<external>>
WebRTC
streams
<<system>>
Signal Android
<<external>>
Google Cloud Messaging
<<external>>
Amazon Route 53
server DNS lookup
sends/ fetch msg,
finds contacts,
exchanges public key
notifies
uses
Signal Foundation
Signal Protocol
User Database
gets user data
Signal Server

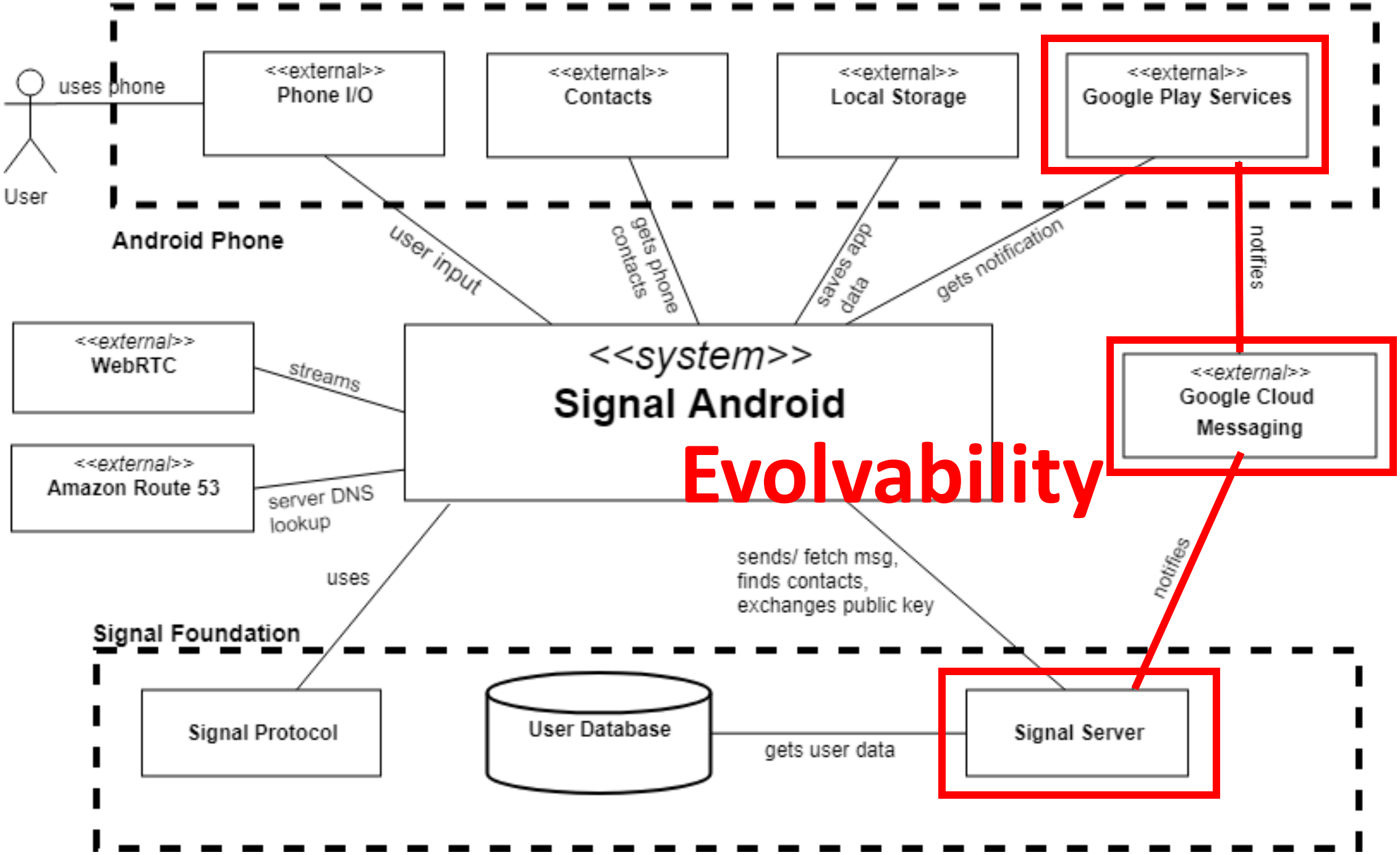

uses phone
User
<<external>>
Phone I/O
<<external>>
Contacts
<<external>>
Local Storage
<<external>>
Google Play Services
Android Phone
user input
gets phone contacts
saves app data
gets notification
notifies
<<external>>
WebRTC
streams
<<system>>
Signal Android
<<external>>
Google Cloud Messaging
<<external>>
Amazon Route 53
server DNS lookup
uses
sends/ fetch msg,
finds contacts,
exchanges public key
notifies
Signal Foundation
Signal Protocol
User Database
gets user data
Signal Server


uses phone
User
<<external>>
Phone I/O
<<external>>
Contacts
<<external>>
Local Storage
<<external>>
Google Play Services
Android Phone
user input
gets phone contacts
saves app data
gets notification
notifies
<<external>>
WebRTC
streams
<<system>>
Signal Android
<<external>>
Google Cloud Messaging
<<external>>
Amazon Route 53
server DNS lookup
uses
sends/ fetch msg,
finds contacts,
exchanges public key
notifies
Signal Foundation
Signal Protocol
User Database
gets user data
Signal Server

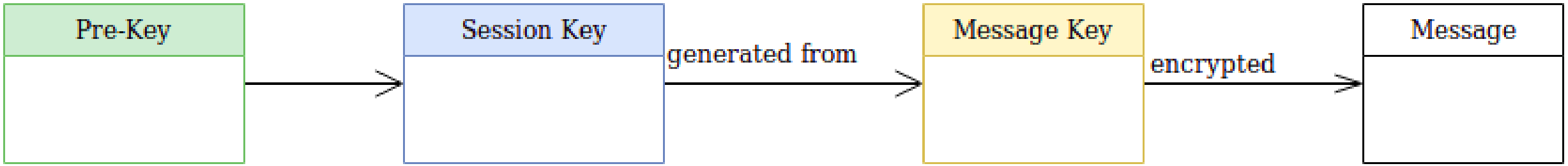

uses phone
User
<<external>>
Phone I/O
<<external>>
Contacts
<<external>>
Local Storage
<<external>>
Google Play Services
Android Phone
user input
gets phone contacts
saves app data
gets notification
notifies
<<external>>
WebRTC
streams
<<system>>
Signal Android
<<external>>
Google Cloud Messaging
<<external>>
Amazon Route 53
server DNS lookup
uses
sends/ fetch msg,
finds contacts,
exchanges public key
notifies
Signal Foundation
Signal Protocol
User Database
gets user data
Signal Server


uses phone
User
<<external>>
Phone I/O
<<external>>
Contacts
<<external>>
Local Storage
<<external>>
Google Play Services
Android Phone
user input
gets phone contacts
saves app data
gets notification
notifies
<<external>>
WebRTC
streams
<<system>>
Signal Android
<<external>>
Google Cloud Messaging
<<external>>
Amazon Route 53
server DNS lookup
uses
sends/ fetch msg,
finds contacts,
exchanges public key
notifies
privacy
Signal Foundation
Signal Protocol
User Database
gets user data
Signal Server


uses phone
User
<<external>>
Phone I/O
<<external>>
Contacts
<<external>>
Local Storage
<<external>>
Google Play Services
Android Phone
user input
gets phone contacts
saves app data
gets notification
notifies
<<external>>
WebRTC
streams
<<system>>
Signal Android
<<external>>
Google Cloud Messaging
<<external>>
Amazon Route 53
server DNS lookup
uses
sends/ fetch msg,
finds contacts,
exchanges public key
notifies
Signal Foundation
Signal Protocol
User Database
gets user data
Signal Server


uses phone
User
<<external>>
Phone I/O
<<external>>
Contacts
<<external>>
Local Storage
<<external>>
Google Play Services
Android Phone
user input
gets phone contacts
saves app data
gets notification
notifies
<<external>>
WebRTC
streams
<<system>>
Signal Android
<<external>>
Google Cloud Messaging
<<external>>
Amazon Route 53
server DNS lookup
Evolvability
uses
sends/ fetch msg,
finds contacts,
exchanges public key
notifies
Signal Foundation
Signal Protocol
User Database
gets user data
Signal Server

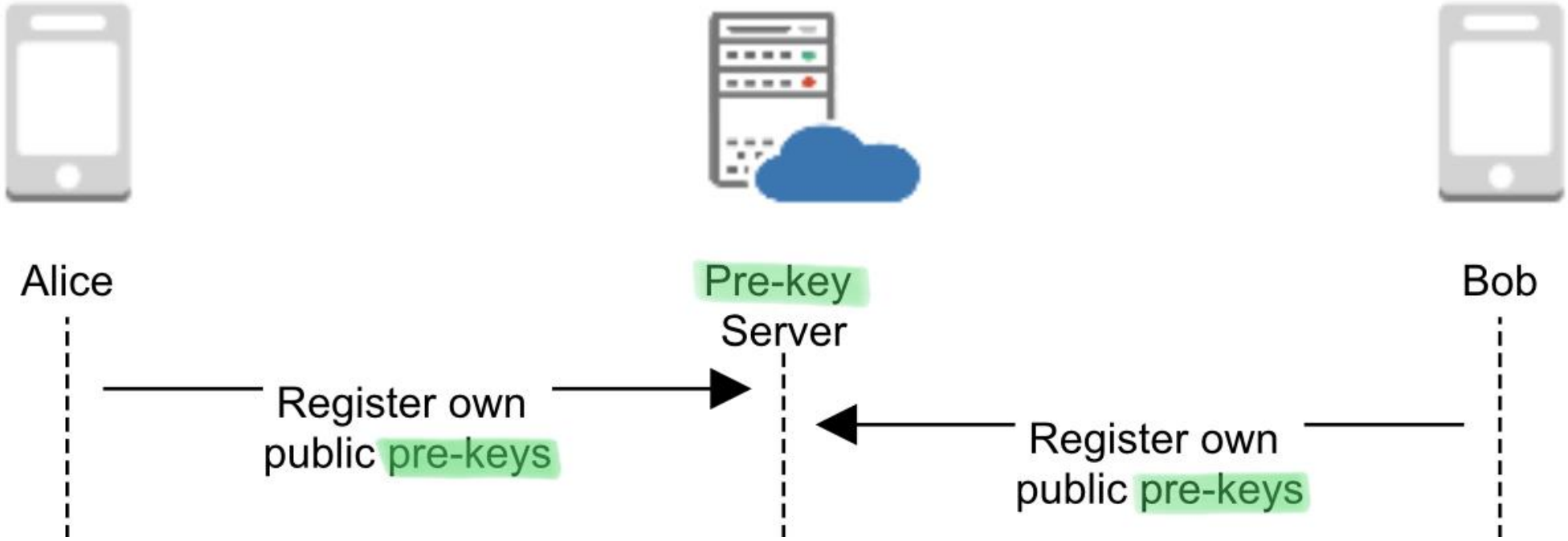
Key types

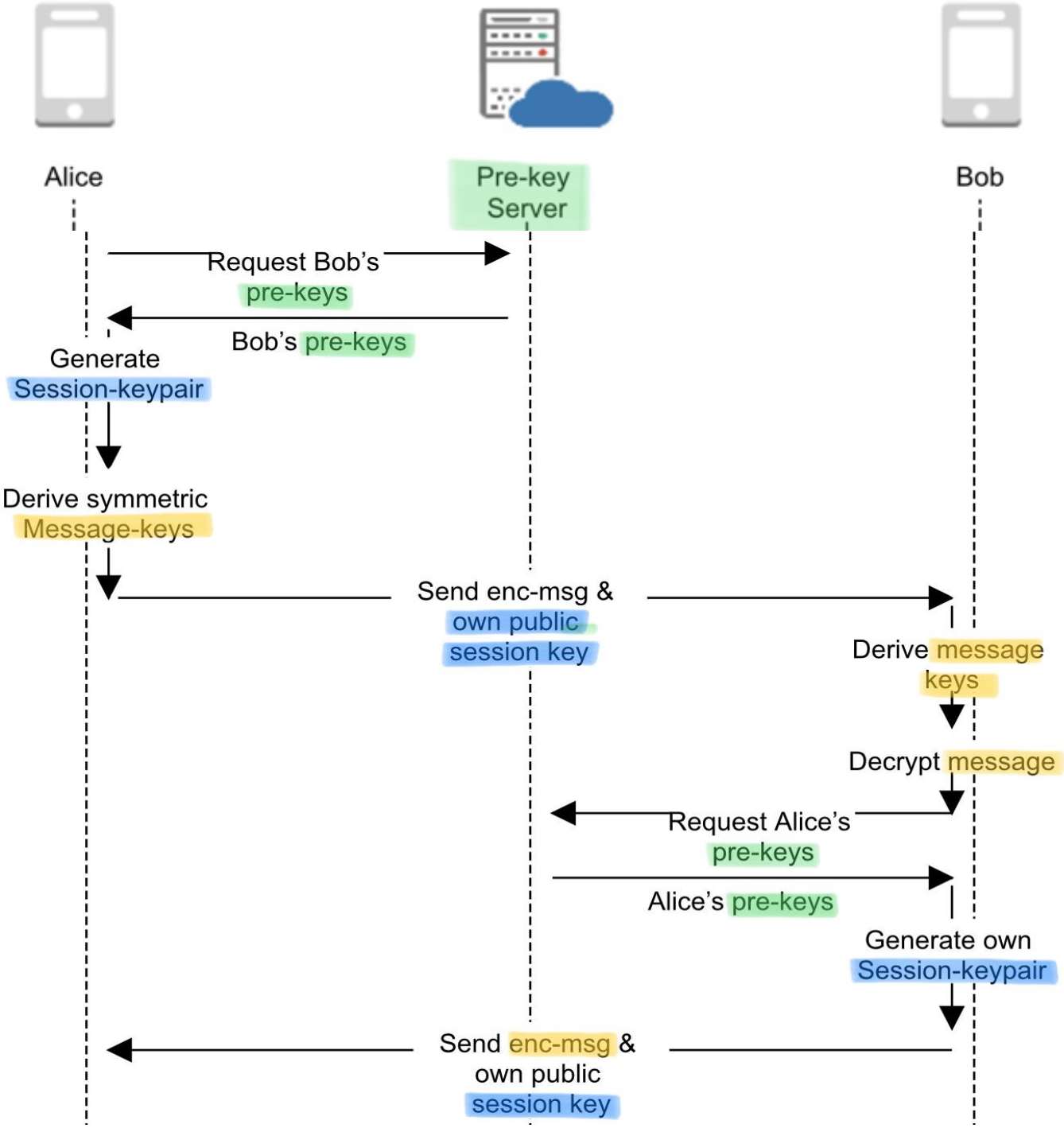

Pre-Key
Session Key
generated from
Message Key
encrypted
Message

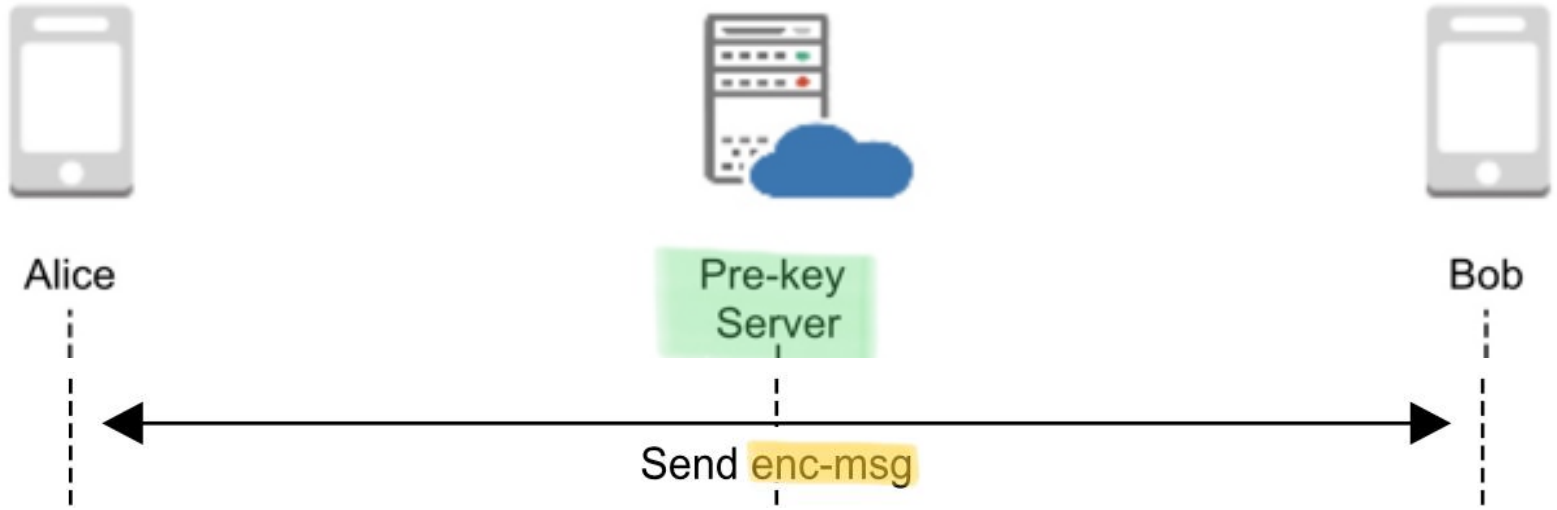
Key exchange

Three stages:

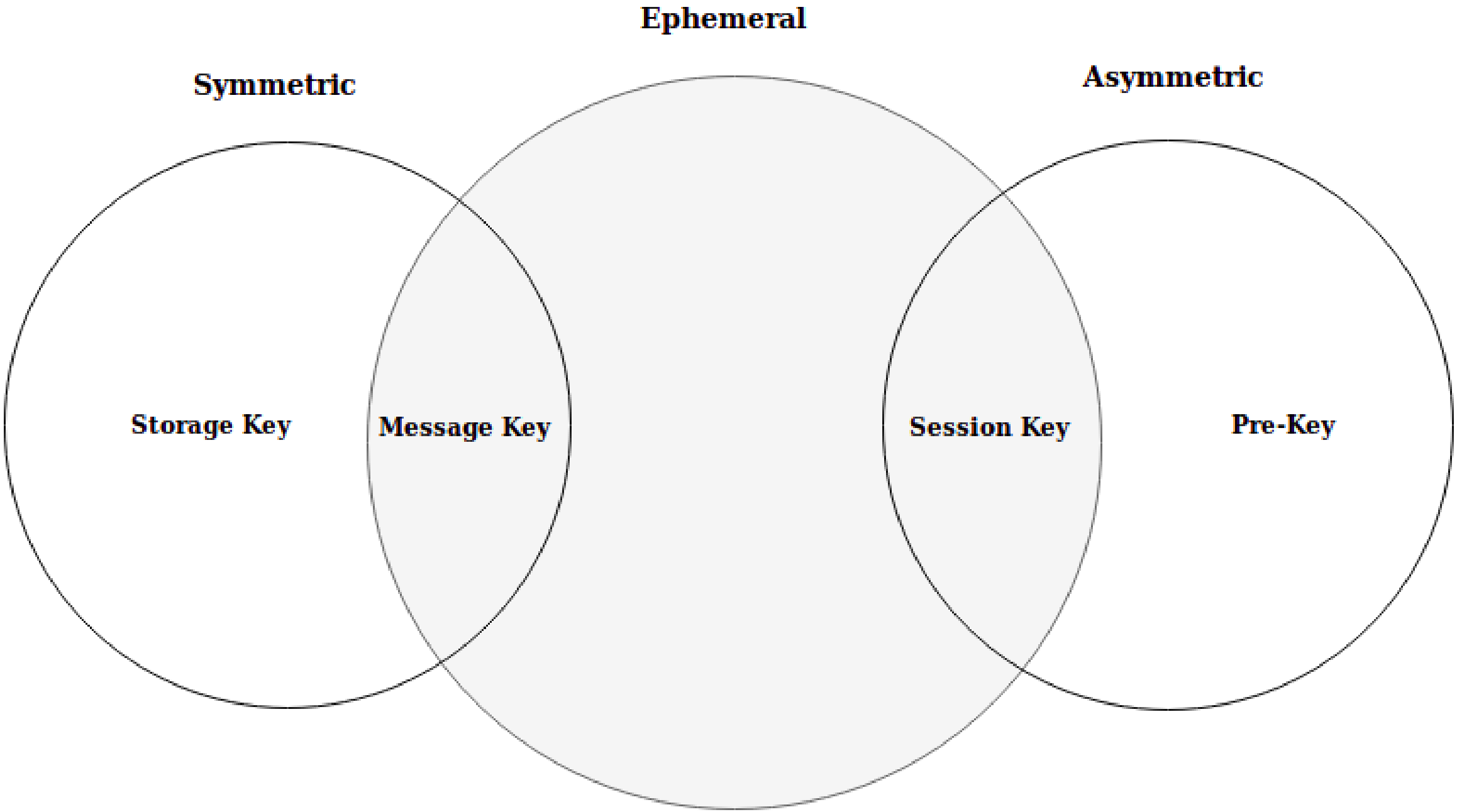
- Register users
- Establish session between two users
- Exchange encrypted messages

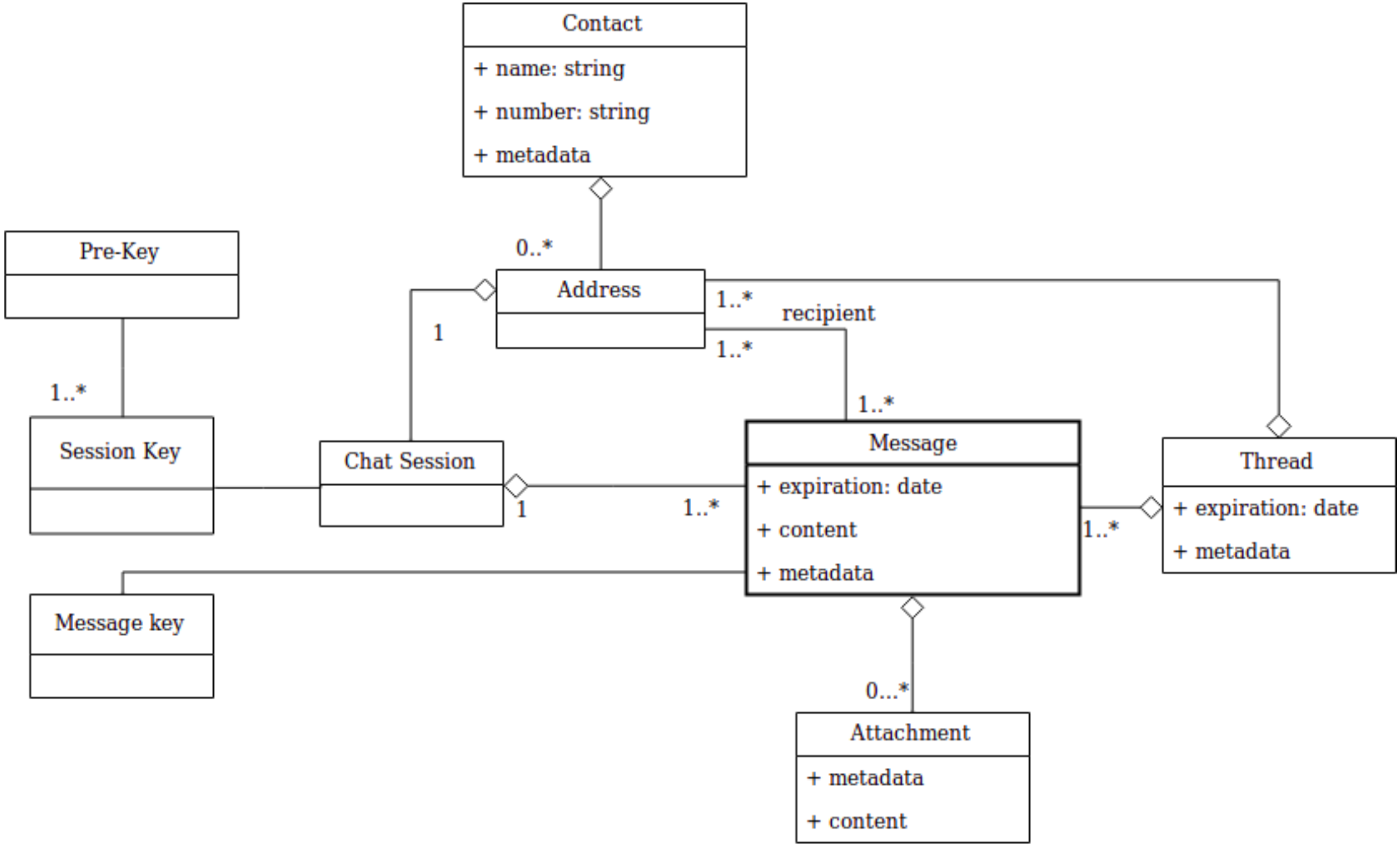



Alice
Pre-key Server
Bob
Request Bob's pre-keys
Bob's pre-keys
Generate Session-keypair
Derive symmetric Message-keys
Send enc-msg & own public session key
Derive message keys
Decrypt message
Request Alice's pre-keys
Alice's pre-keys
Generate own Session-keypair
Send enc-msg & own public session key

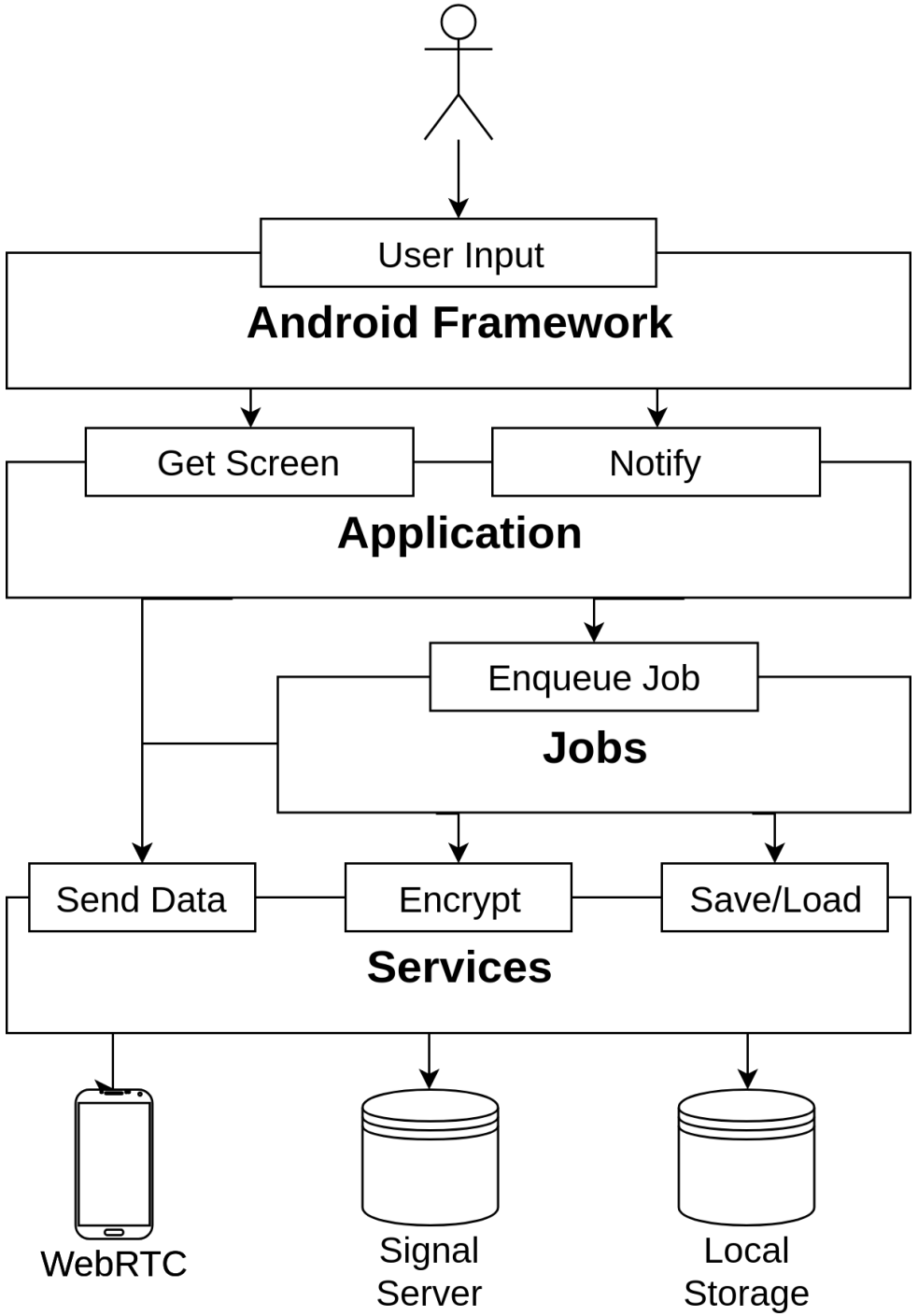


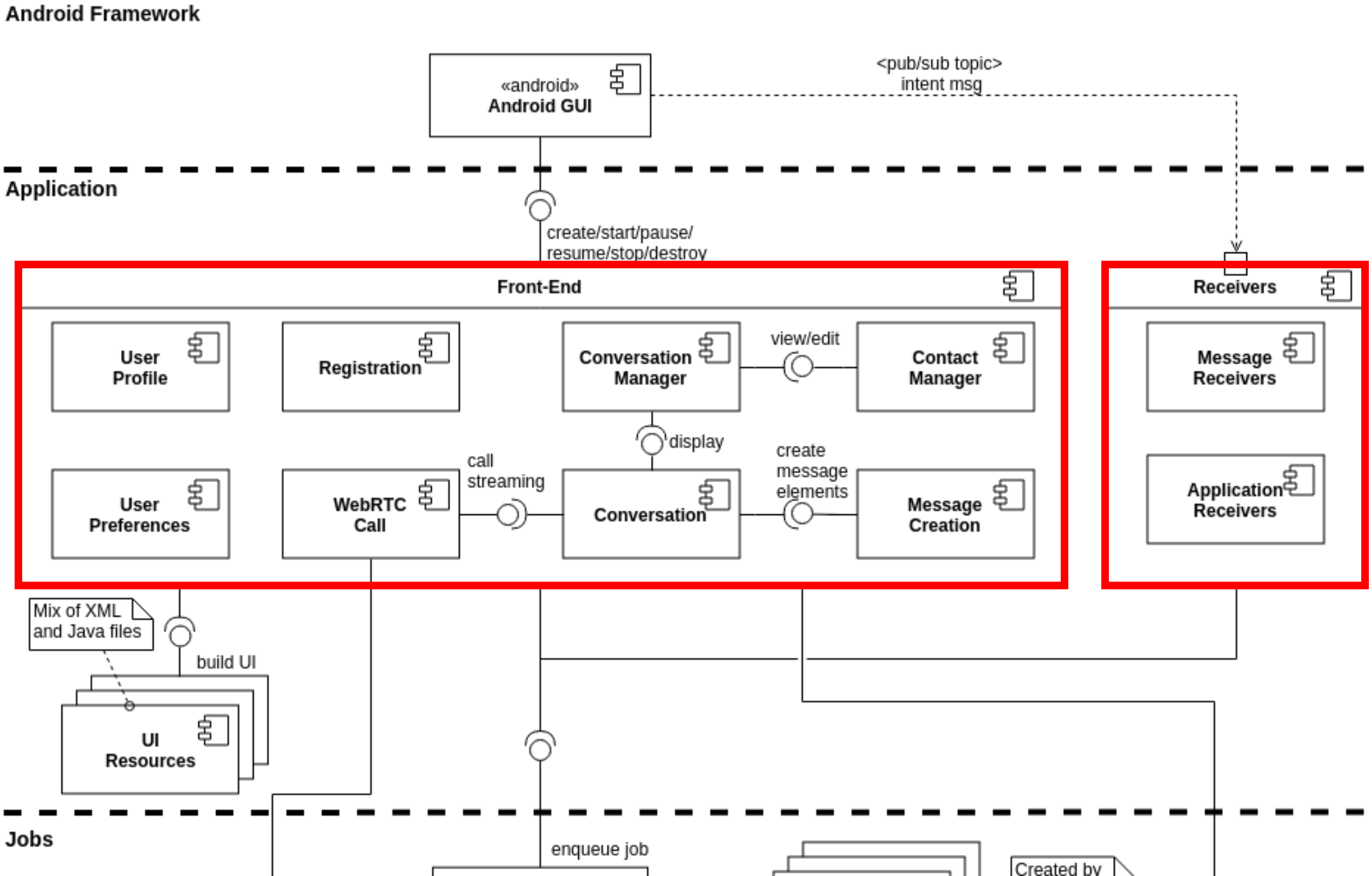
- Recap:
  - **Pre-keys, one per user**
    Each user uploads his pre-key at registration.
  - **Session keys, one pair per session.**
    Generated for a user-to-user conversation.
  - **Message keys, one per message.**
    Derived from Session keys.


Ephemeral
Symmetric
Asymmetric
Storage Key
Message Key
Session Key
Pre-Key

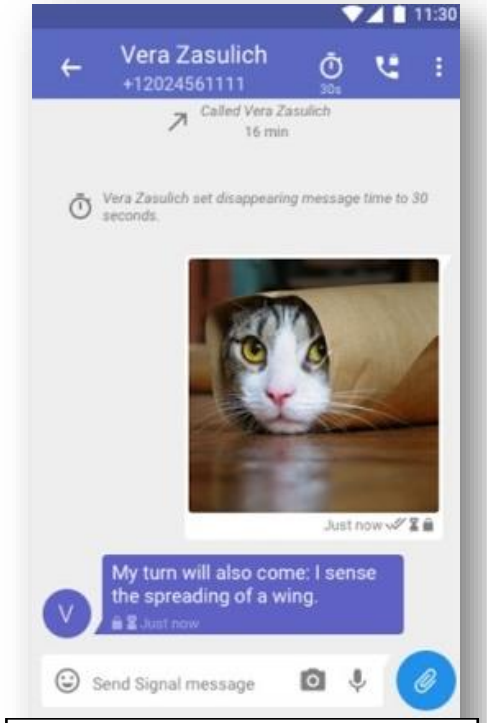

Contact
+ name: string
+ number: string
+ metadata
0..*
Address
1
1..*
recipient
1..*
Pre-Key
1..*
Session Key
Chat Session
1
1..*
Message
+ expiration: date
+ content
+ metadata
1..*
Thread
+ expiration: date
+ metadata
Message key
0...*
Attachment
+ metadata
+ content

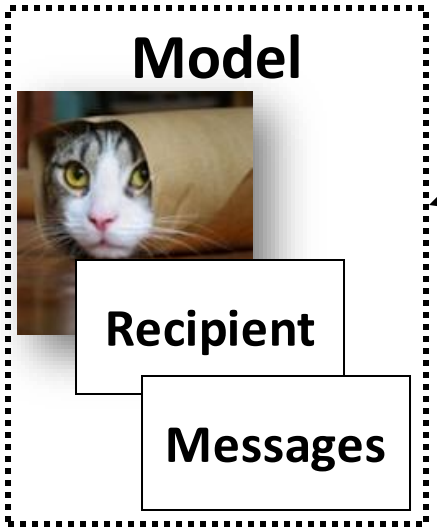
Layered Decomposition




Android Framework
«android»
Android GUI
<pub/sub topic>
intent msg
Application
create/start/pause/
resume/stop/destroy
Front-End
Receivers
User
Profile
Registration
Conversation
Manager
view/edit
Contact
Manager
Message
Receivers
display
call
streaming
create
message
elements
User
Preferences
WebRTC
Call
Conversation
Message
Creation
Application
Receivers
Mix of XML
and Java files
build UI
UI
Resources
Jobs
enqueue job
Created by

Front-End

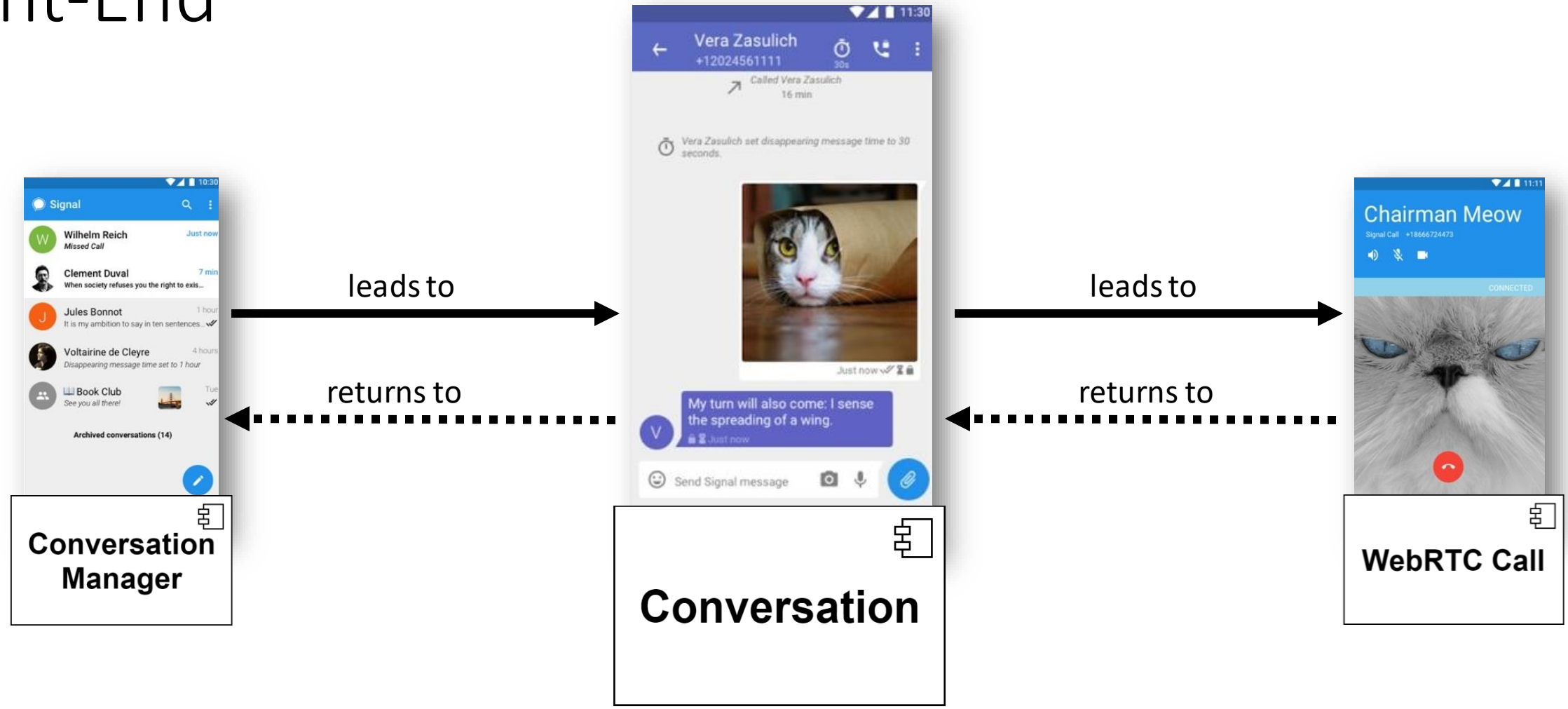


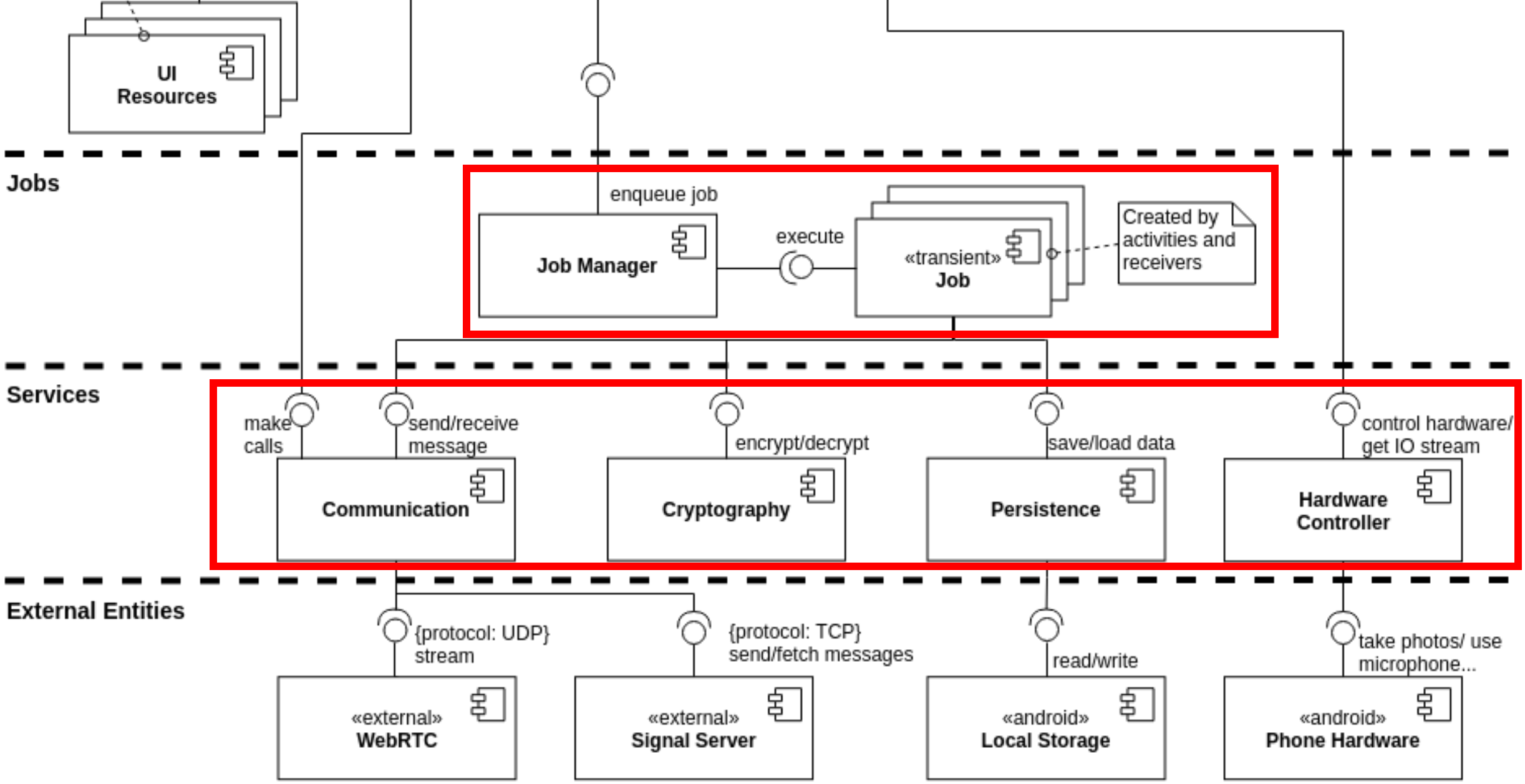


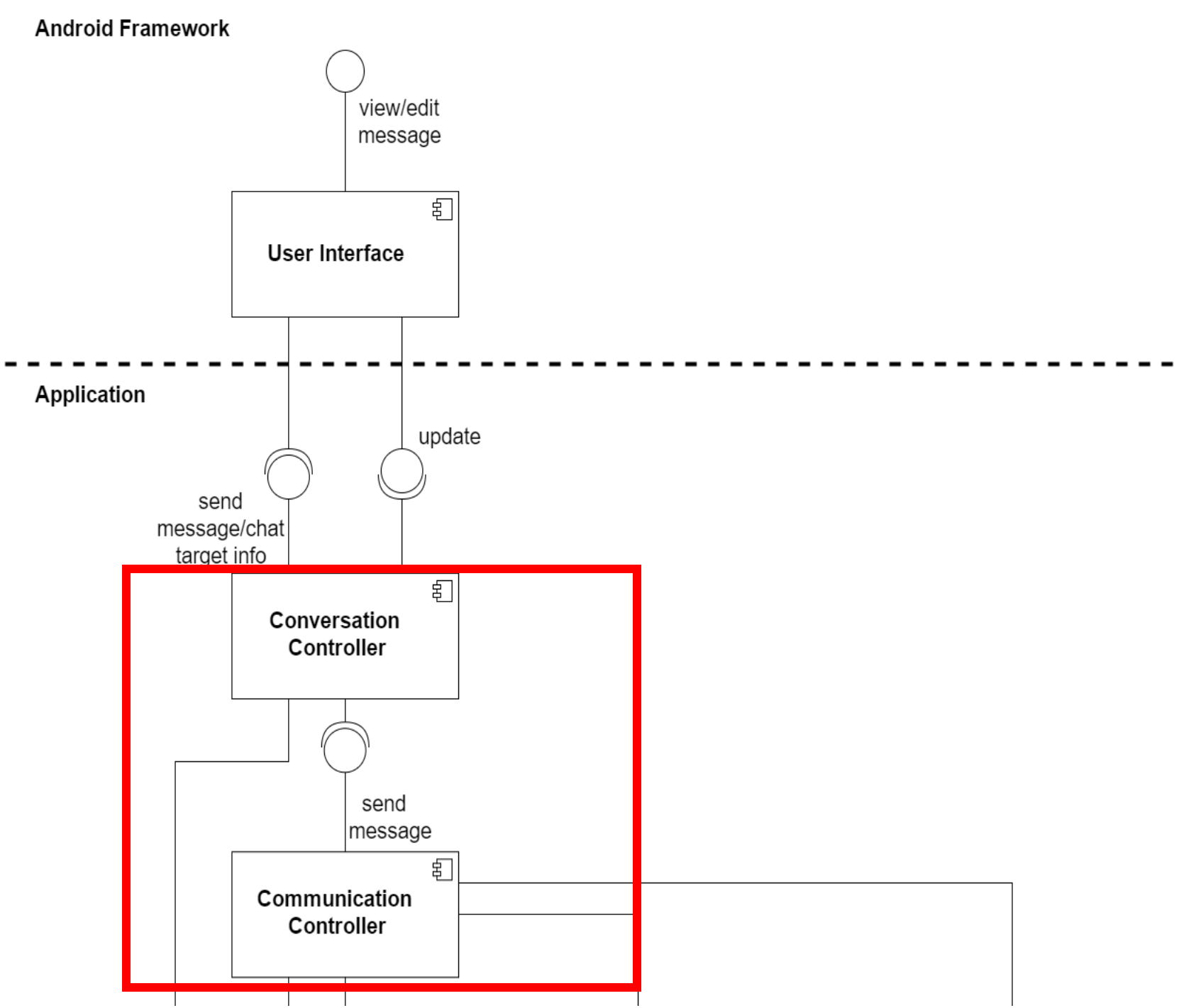
depends on

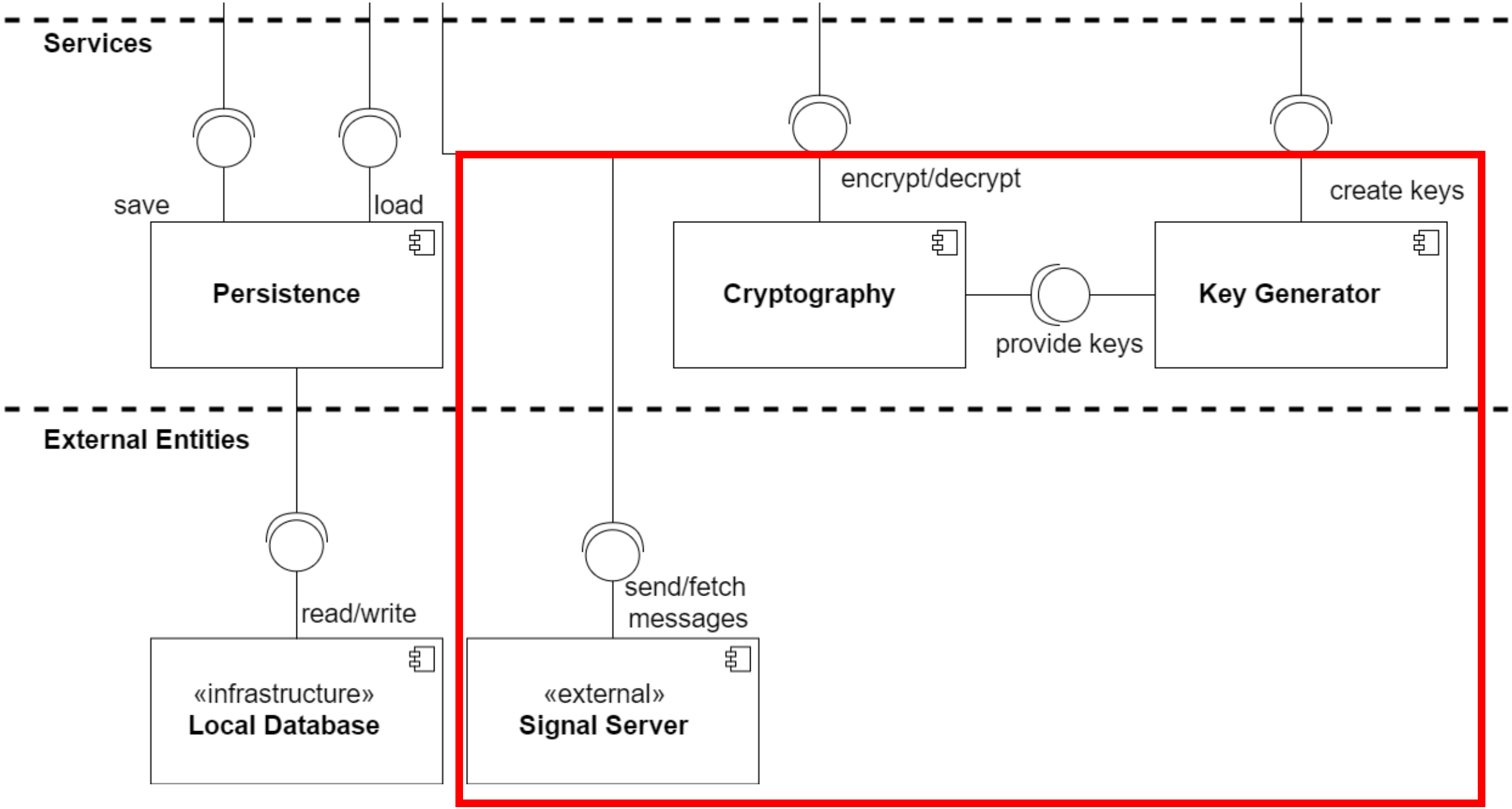
depends on

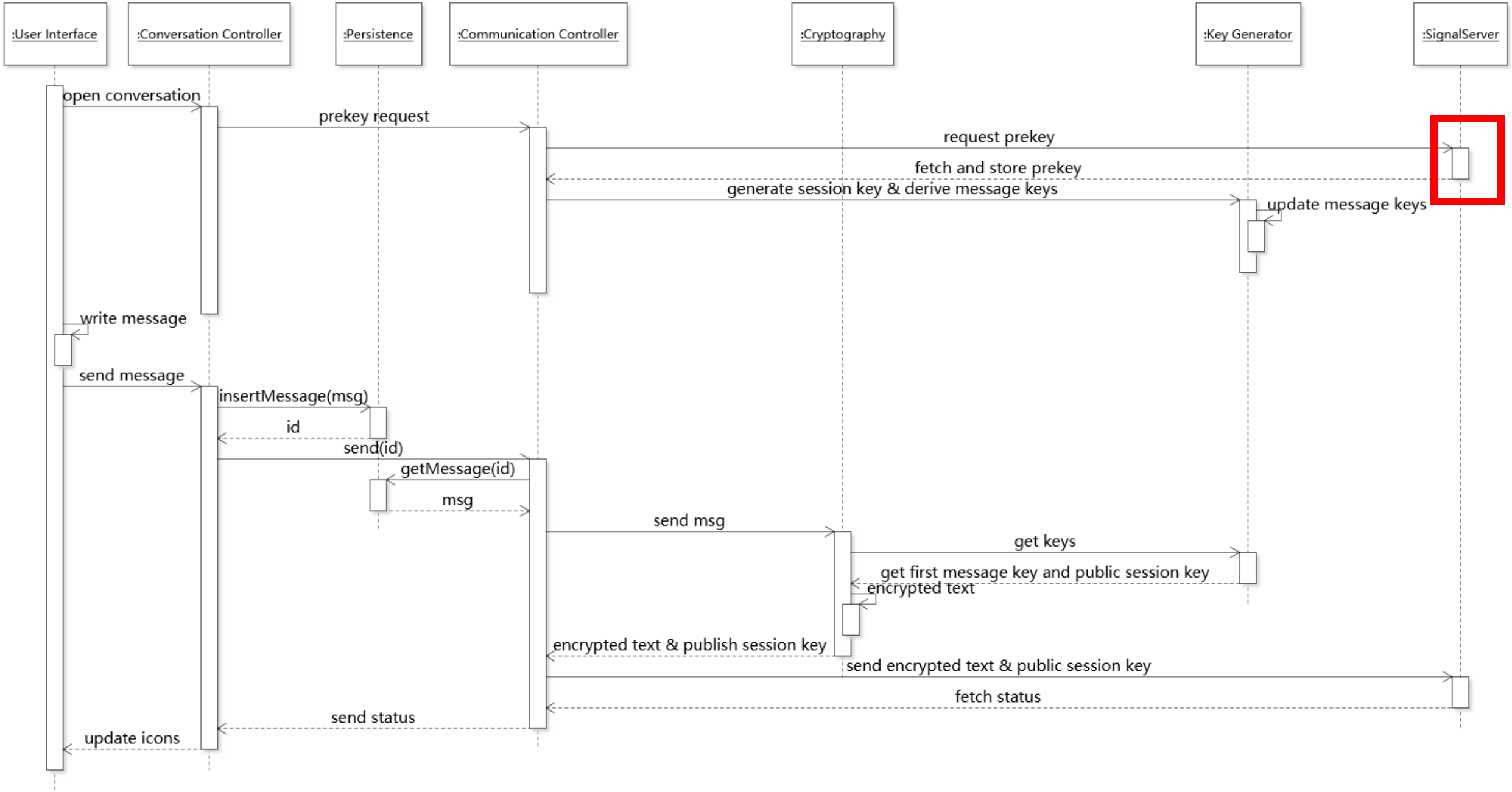
Front-End

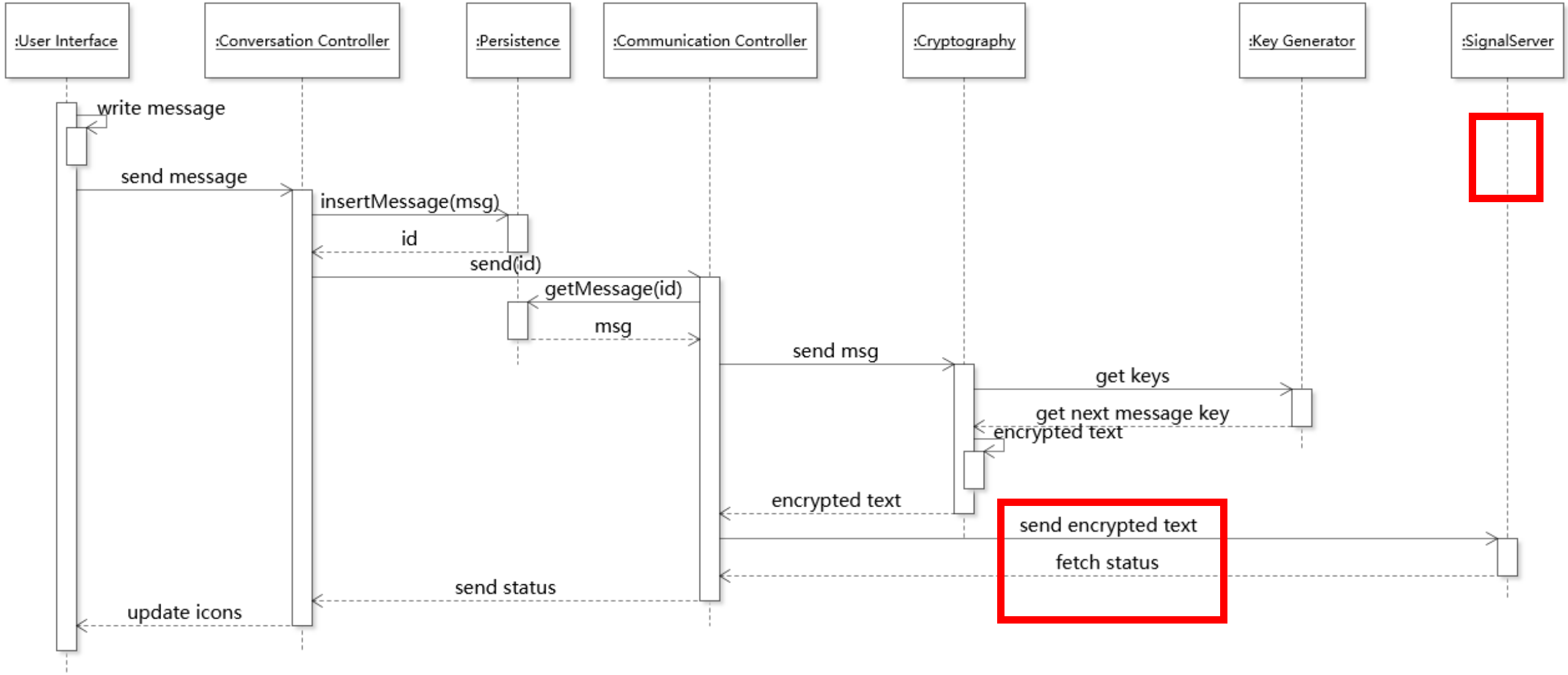

Signal
Wilhelm Reich
Missed Call
Just now
Clement Duval
When society refuses you the right to exis...
7 min
Jules Bonnot
It is my ambition to say in ten sentences...
1 hour
Voltairine de Cleyre
Disappearing message time set to 1 hour
4 hours
Book Club
See you all there!
Tue
Archived conversations (14)
Conversation Manager
leads to
returns to
Vera Zasulich
+12024561111
Called Vera Zasulich
16 min
Vera Zasulich set disappearing message time to 30 seconds.
Just now
My turn will also come: I sense the spreading of a wing.
Just now
Send Signal message
Conversation
leads to
returns to
Chairman Meow
Signal Call +18666724473
CONNECTED
WebRTC Call


UI
Resources
Jobs
enqueue job
Job Manager
execute
«transient»
Job
Created by
activities and
receivers
Services
make
calls
send/receive
message
Communication
encrypt/decrypt
Cryptography
save/load data
Persistence
control hardware/
get IO stream
Hardware
Controller
External Entities
{protocol: UDP}
stream
«external»
WebRTC
{protocol: TCP}
send/fetch messages
«external»
Signal Server
read/write
«android»
Local Storage
take photos/ use
microphone...
«android»
Phone Hardware


Android Framework
view/edit
message
User Interface
Application
update
send
message/chat
target info
Conversation
Controller
send
message
Communication
Controller


Services
save
load
Persistence
encrypt/decrypt
create keys
Cryptography
provide keys
Key Generator
External Entities
read/write
«infrastructure»
Local Database
send/fetch
messages
«external»
Signal Server


:User Interface
:Conversation Controller
:Persistence
:Communication Controller
:Cryptography
:Key Generator
:SignalServer
open conversation
prekey request
request prekey
fetch and store prekey
generate session key & derive message keys
update message keys
write message
send message
insertMessage(msg)
id
send(id)
getMessage(id)
msg
send msg
get keys
get first message key and public session key
encrypted text
encrypted text & publish session key
send encrypted text & public session key
fetch status
send status
update icons


:User Interface
:Conversation Controller
:Persistence
:Communication Controller
:Cryptography
:Key Generator
:SignalServer
write message
send message
insertMessage(msg)
id
send(id)
getMessage(id)
msg
send msg
get keys
get next message key
encrypted text
encrypted text
send encrypted text
fetch status
send status
update icons

Thank you for your attention.