OpenSIPS 2.0
a programmable SIP framework

Bogdan-Andrei Iancu
Founder OpenSIPS Project

What is OpenSIPS ?!

What is OpenSIPS:

- a SIP server....more than a proxy
- doing signaling only....not a media server
- ....GPL, of course
- flexible and powerful in the same time

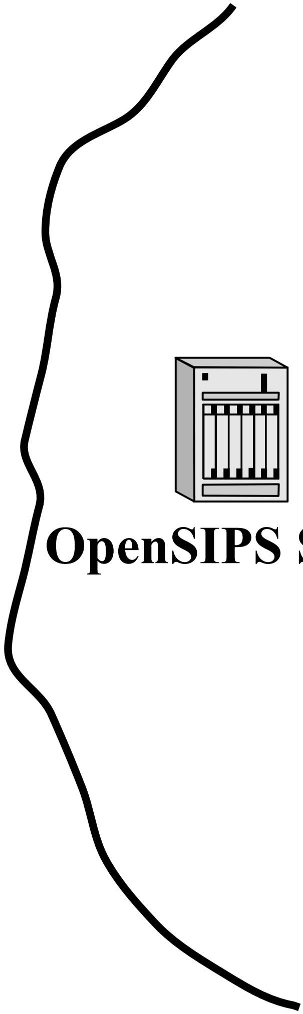
Where to use OpenSIPS ?

- more or less everywhere

Where to use OpenSIPS

Wild Internet

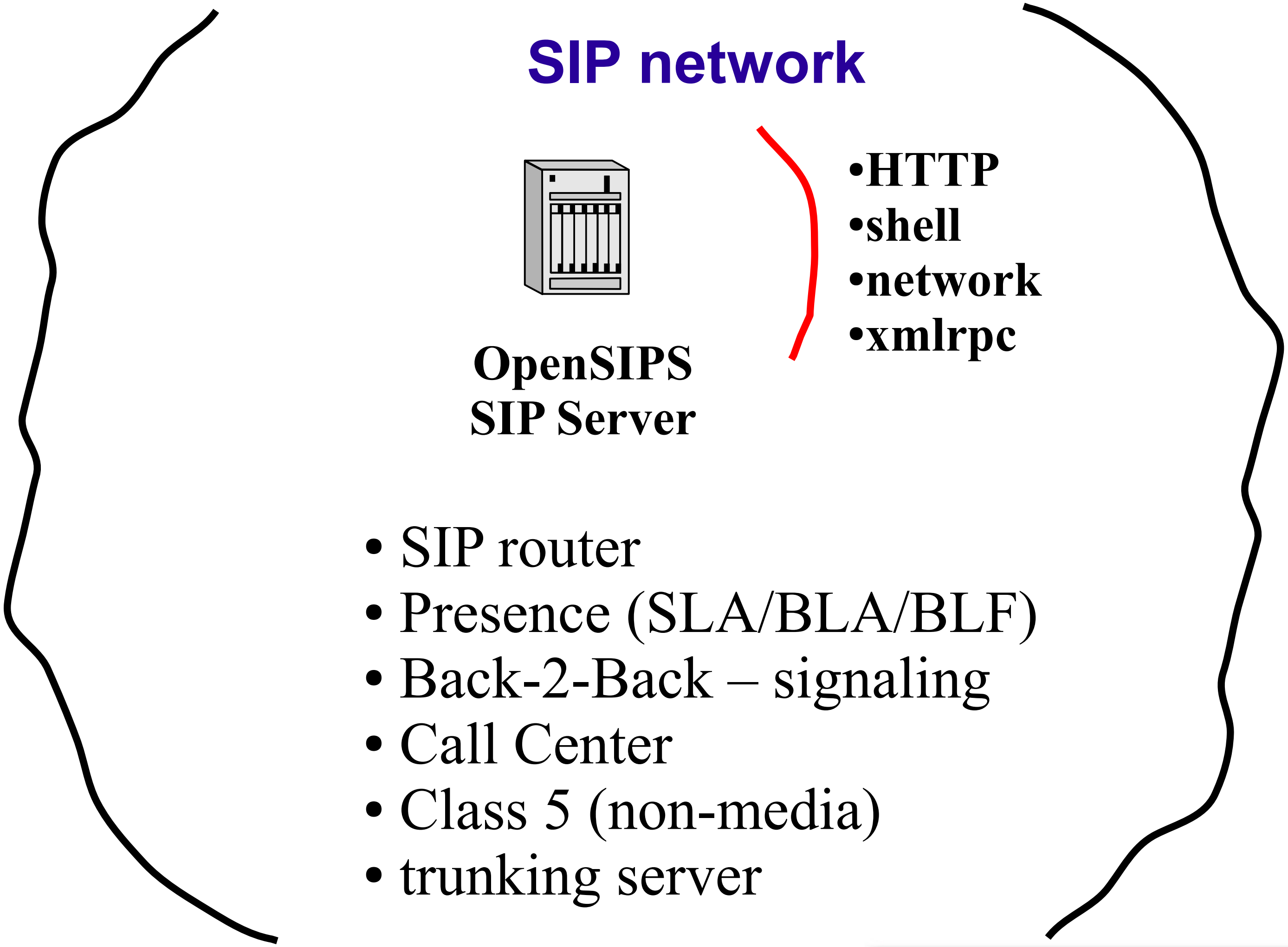
SIP network



OpenSIPS SBC

- nat traversal
- security filter
- SIP validation
- Load balancer
- dialog aware
- HA

Where to use OpenSIPS



SIP network

OpenSIPS
SIP Server

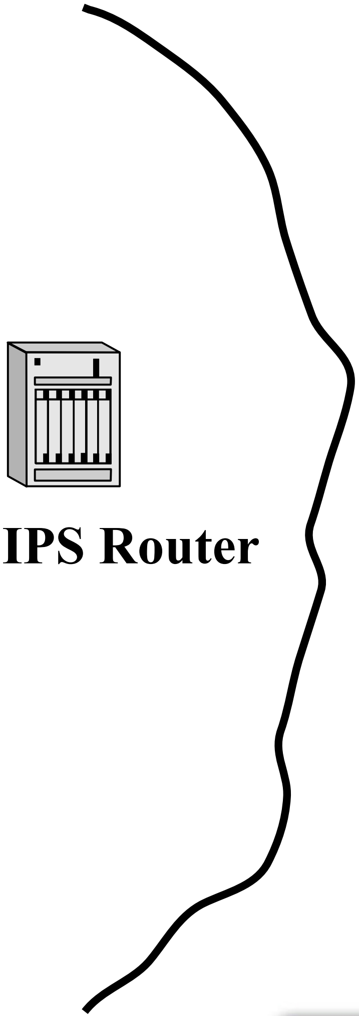
- HTTP
- shell
- network
- xmlrpc

- SIP router
- Presence (SLA/BLA/BLF)
- Back-2-Back – signaling
- Call Center
- Class 5 (non-media)
- trunking server

Where to use OpenSIPS

SIP network

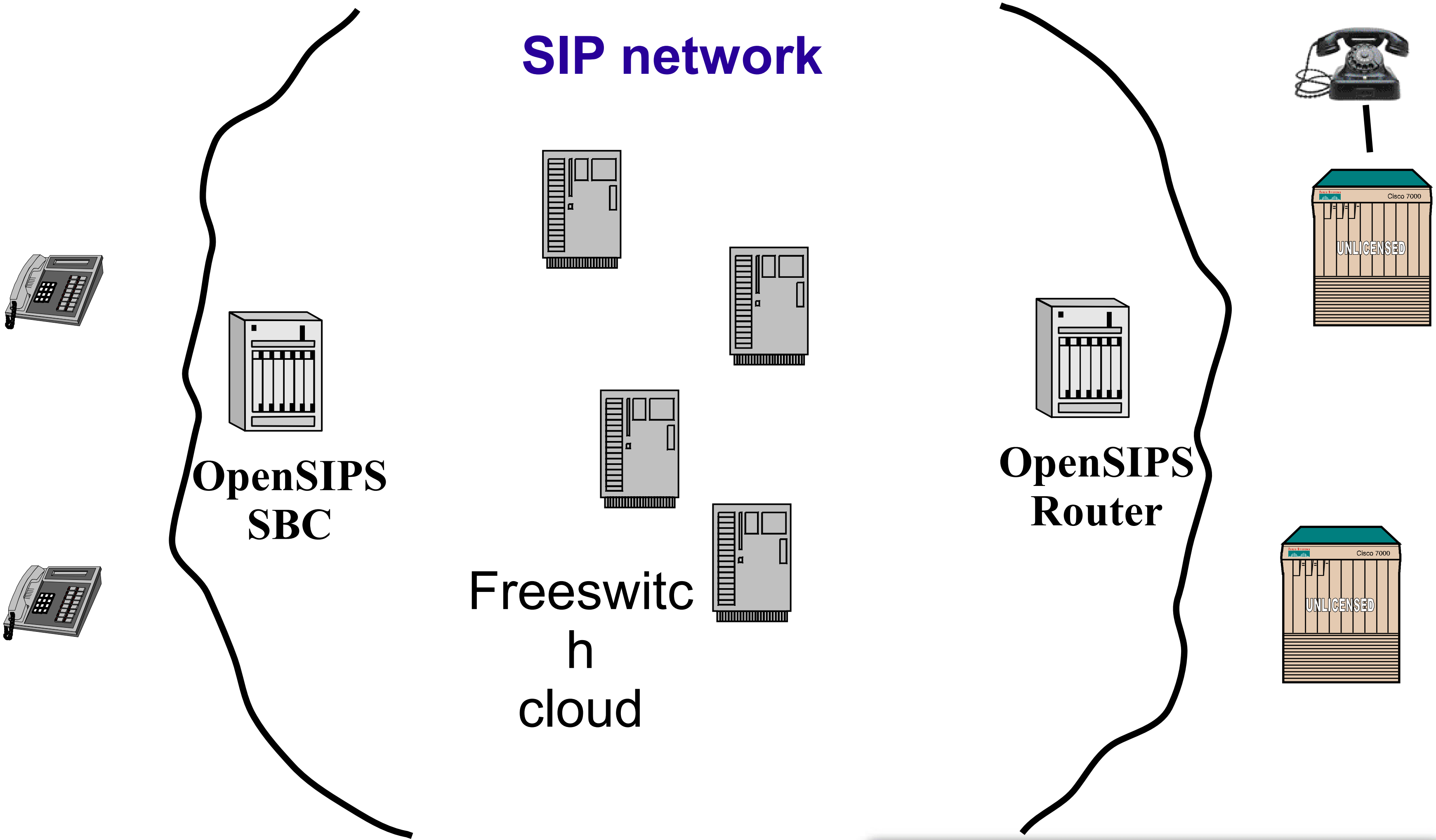
- LCR
- prefix routing
- LNP server
- GW controller
- Topology Hiding



OpenSIPS Router

Wild Internet

The “OpenSIPS Sandwich”



SIP network

OpenSIPS
SBC

Freeswitc
h
cloud

OpenSIPS
Router

OpenSIPS DVD

- a free VM show case of a simple provider setup
- ready to run SIP platform
- proof-of-concept o how to build a SIP platform
- reduces the learning curve

The Story

The Story

Once upon a time there was a powerful and flexible SIP Server ....

....there was OpenSIPS doing tens of thousands CPS.

BUT....

What to be addressed

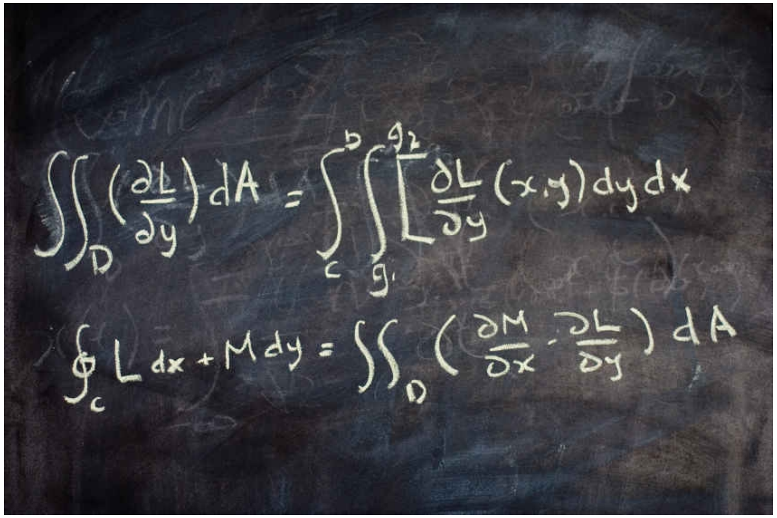
SIP Low level awareness

- you still need to be aware of and handle low level SIP bits and pieces (transactions, dialogs, NAT, etc) to make it work
- you cannot focus only on service creation

What to be addressed

Configuration skills

- **you are required to learn the custom OpenSIPS scripting language**
- **you are limited to what OpenSIPS script language has to offer**
- **the script language is not integration friendly**


$\iint_D \left(\frac{\partial L}{\partial y}\right) dA = \int_c^b \int_{g_1}^{g_2} \left[\frac{\partial L}{\partial y}(x,y) dy dx$
$\oint_C L\,dx + M\,dy = \iint_D \left(\frac{\partial M}{\partial x} - \frac{\partial L}{\partial y}\right) dA$

What to be addressed

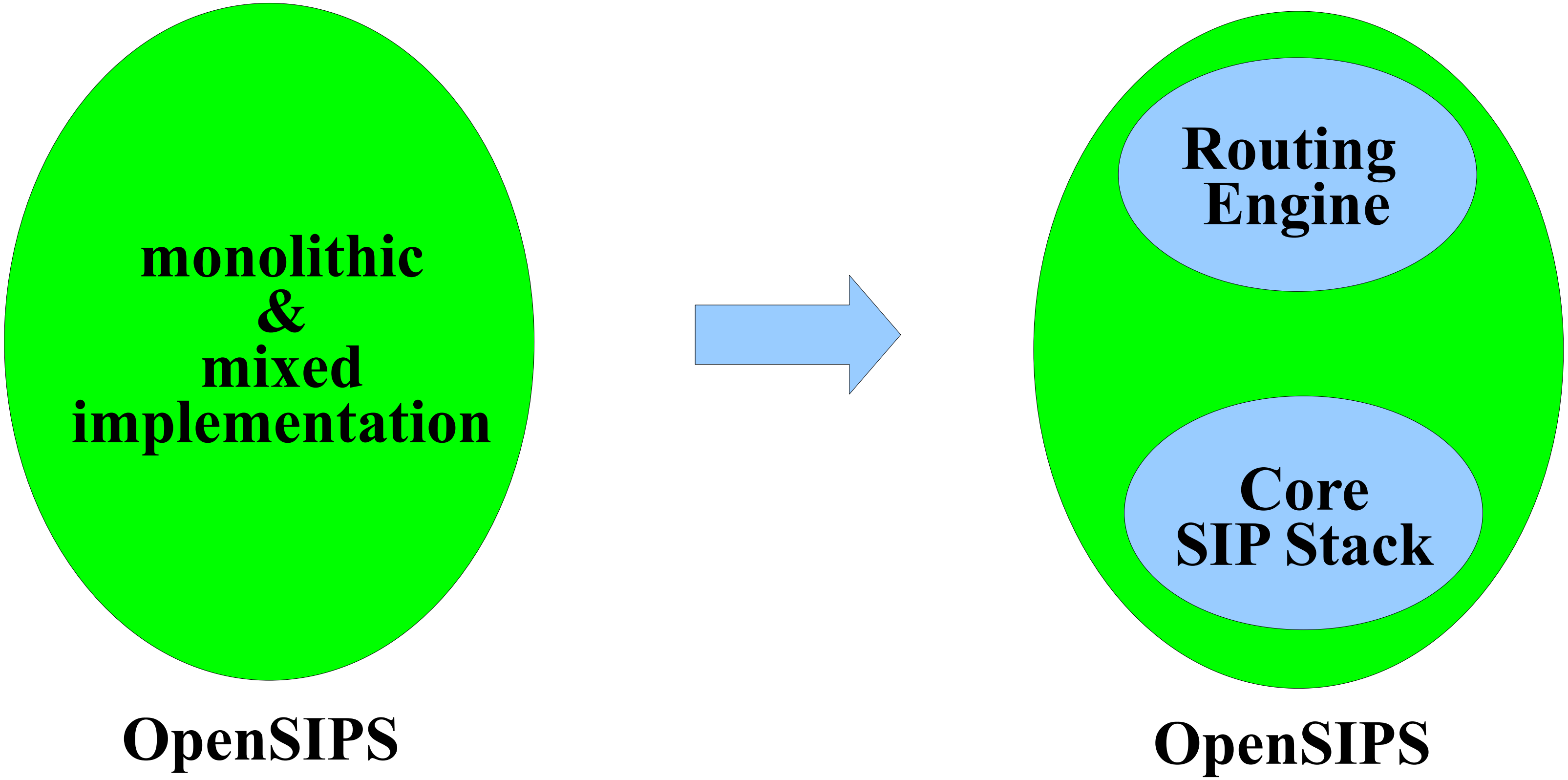
Horizontal Scalability

- cannot scale with a single instance, no matter how powerful it is
- clustering must be naturally achieved
- traffic and data sharing across all nodes in cluster

OpenSIPS 2.0 == 42

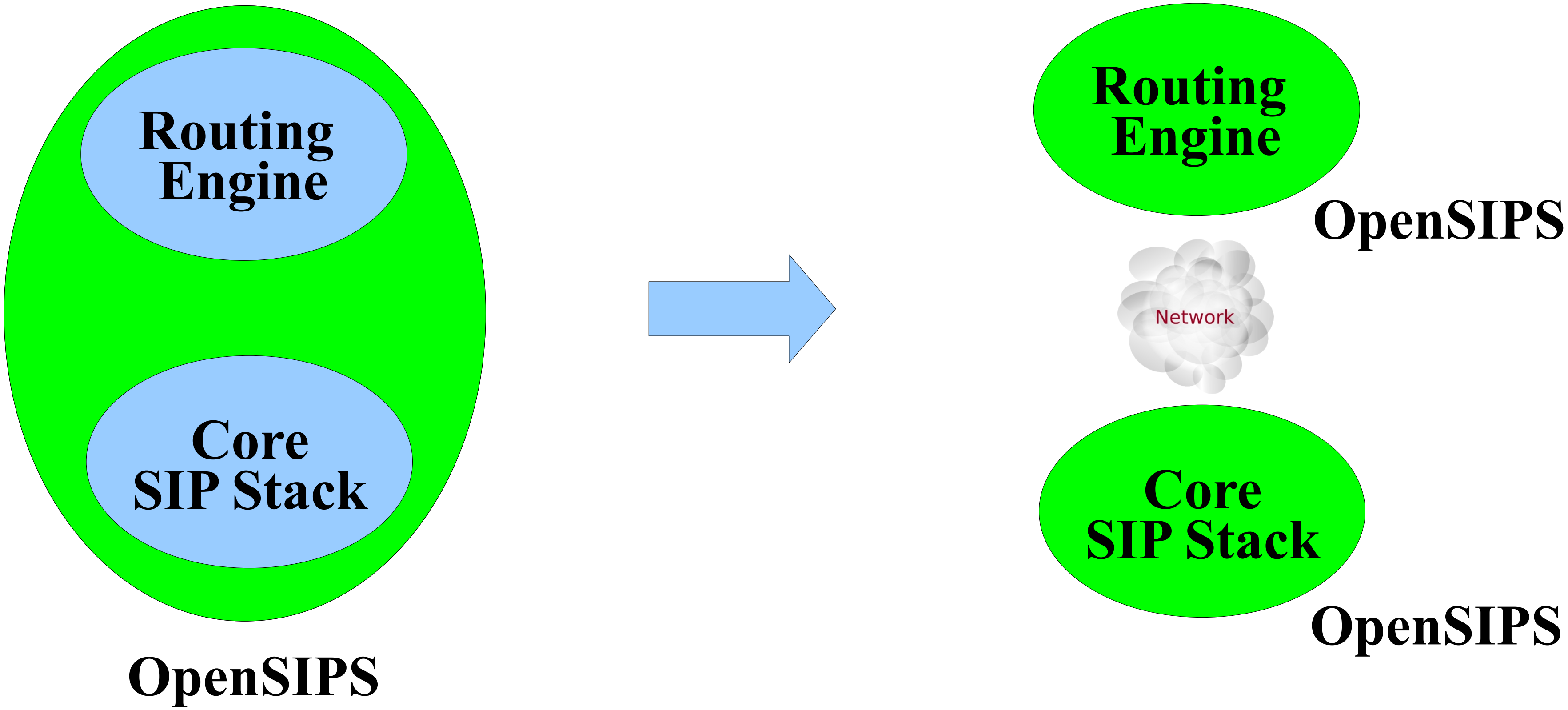
2.0 Genesis

STEP 1 – separation of SIP stack and routing logic



2.0 Genesis

STEP 2 – decouple routing & core to get separated & independent applications

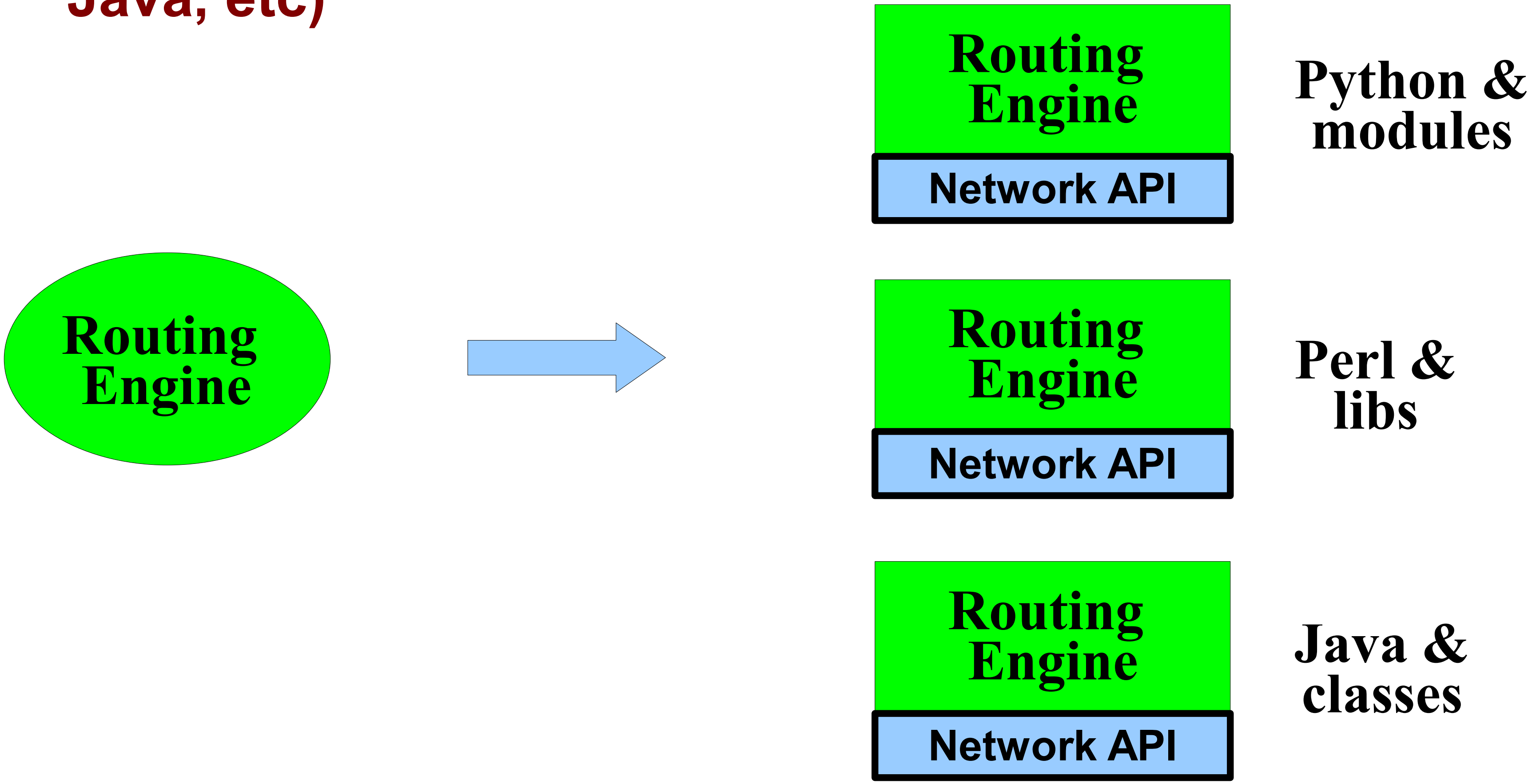


Solves:

- no need to deal SIP low level, just to control and interact with it
- you can focus on service creation without taking care of SIP specific details
- achieve vertical scalability (routing logic and core may be on different machines)
- optimize the processing – the Core part (SIP stack) may automatically and transparent handle certain SIP events (like CANCEL, ACK, etc)
- core is based on an ***asynchronous reactor*** to avoid blocking during I/O ops (TCP, DB, DNS, etc)

2.0 Genesis

STEP 3 – programmable routing logic (Perl, Python, Java, etc)

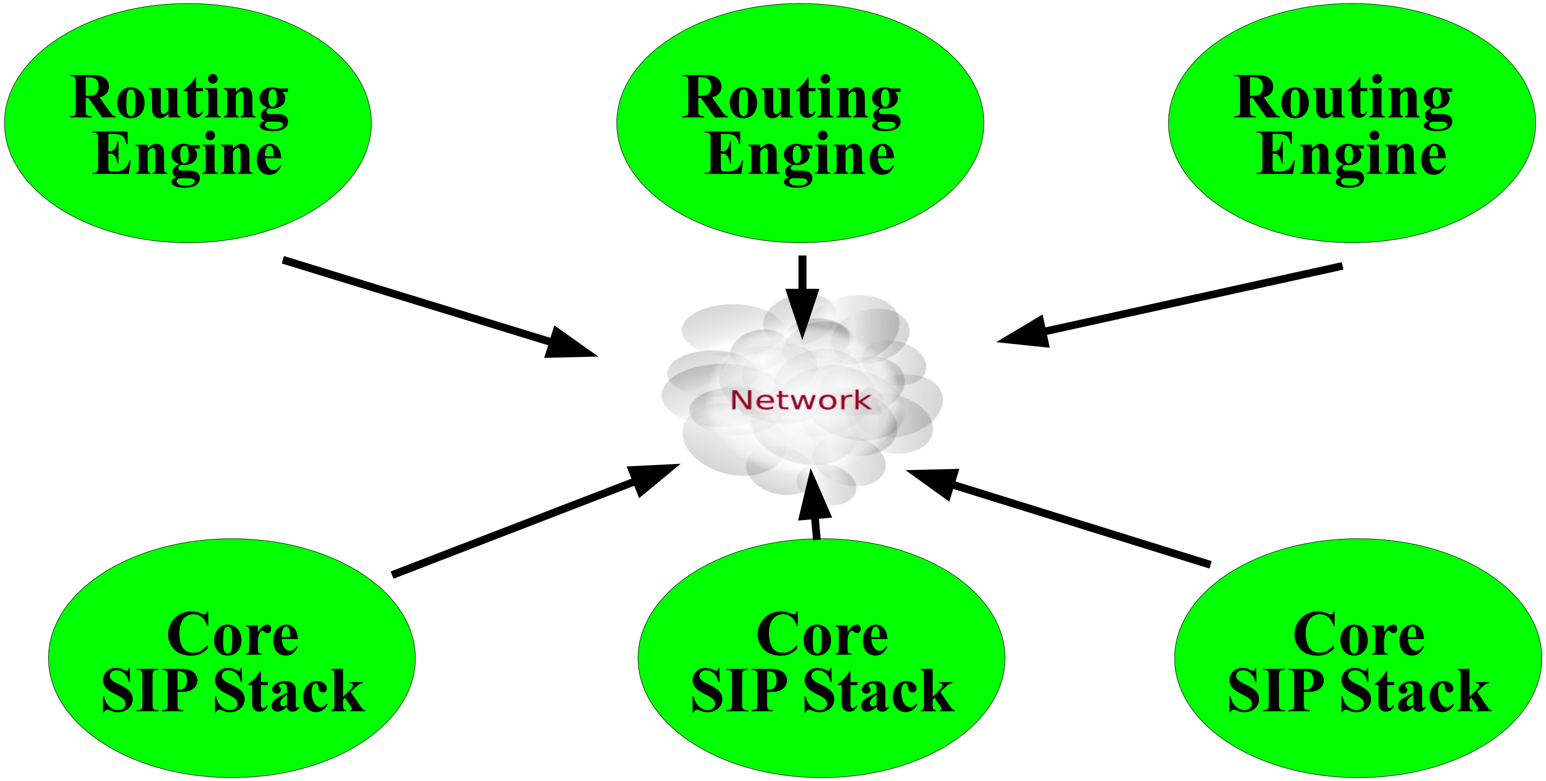


Solves:

- no more custom language for scripting – you can use your own favorite language (any)
- scripting is no more limited – you can take full advantages of the capabilities (as scripting) and already existing functionality from the ligh-level programming languages
- integration (with whatever other apps in whatever other languages) become trivial as what language is used is no more a limitation, but rather an advantage
- routing logic can be actually part of other larger application

2.0 Genesis

STEP 4 – horizontal scalability for both Core and Logic



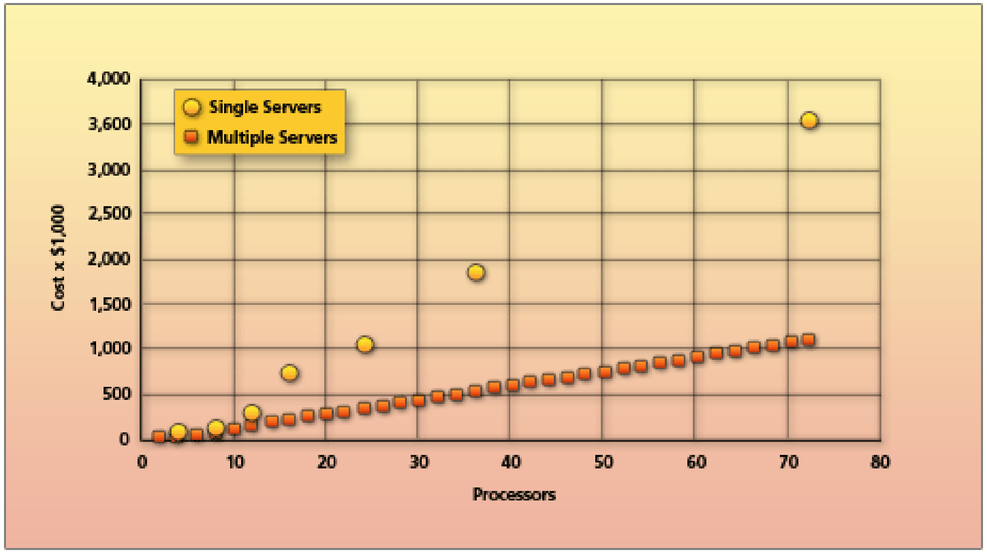
2/1/11

Solves:

- horizontal scalability – each part (core and logic) may individually scale across several machines.

- the logic will be responsible for clustering (service and data) by providing to Core part data storage support

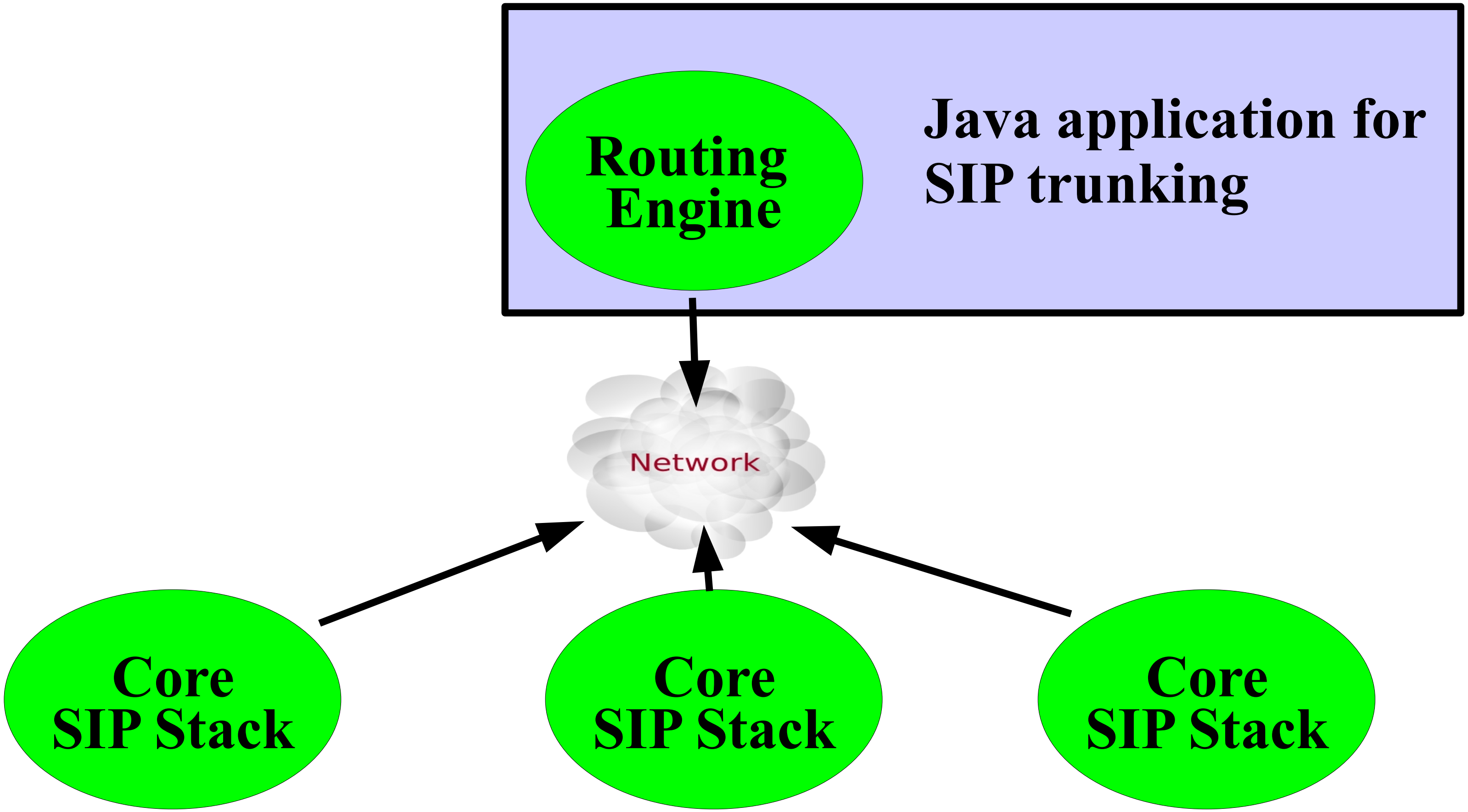
- it is cheaper to scale (for same number of CPUs) with several machines, rather than only one



Examples

Examples

Intensive traffic, simple logic



Routing Engine

Java application for SIP trunking

Network

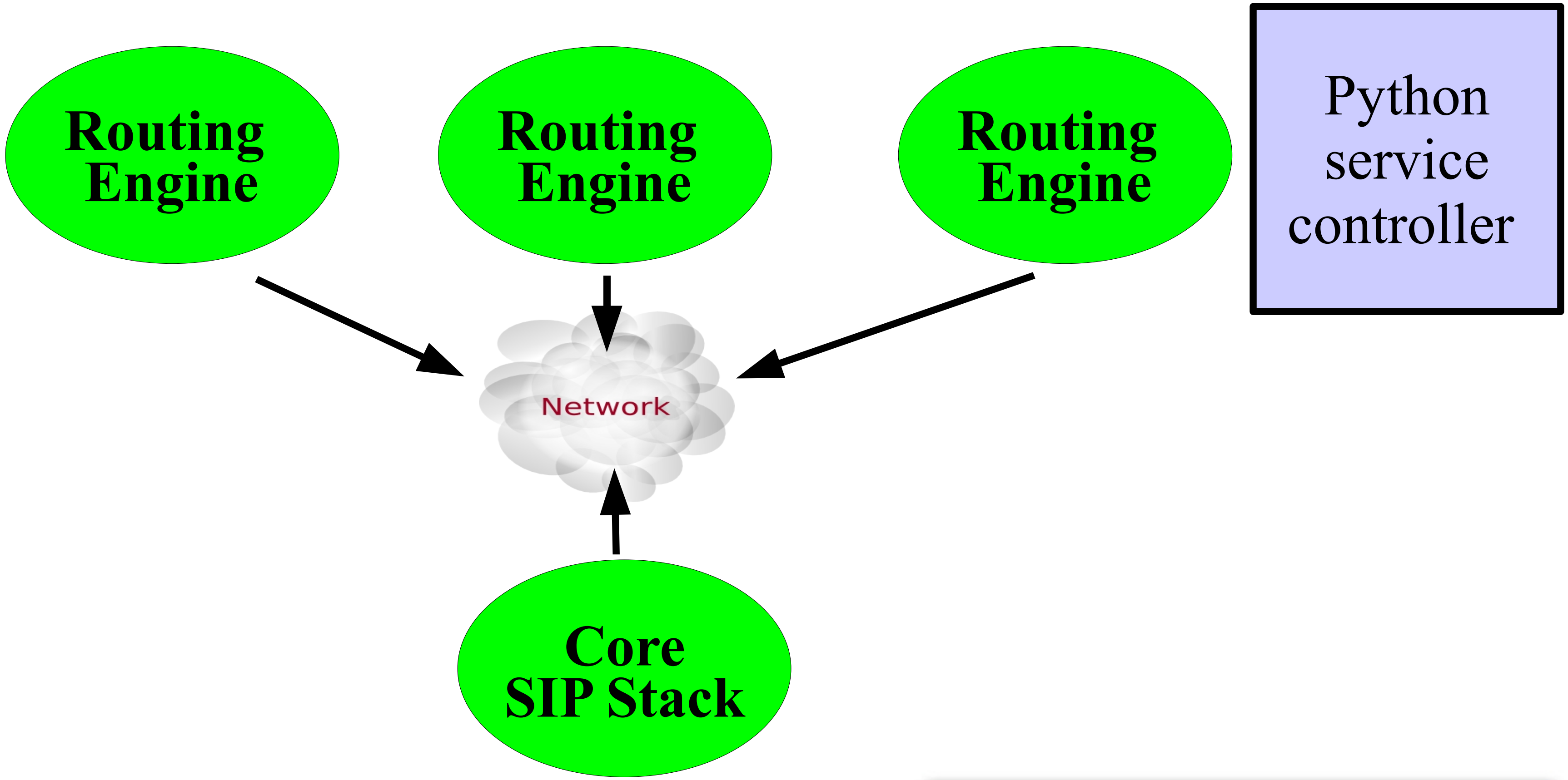
Core SIP Stack

Core SIP Stack

Core SIP Stack

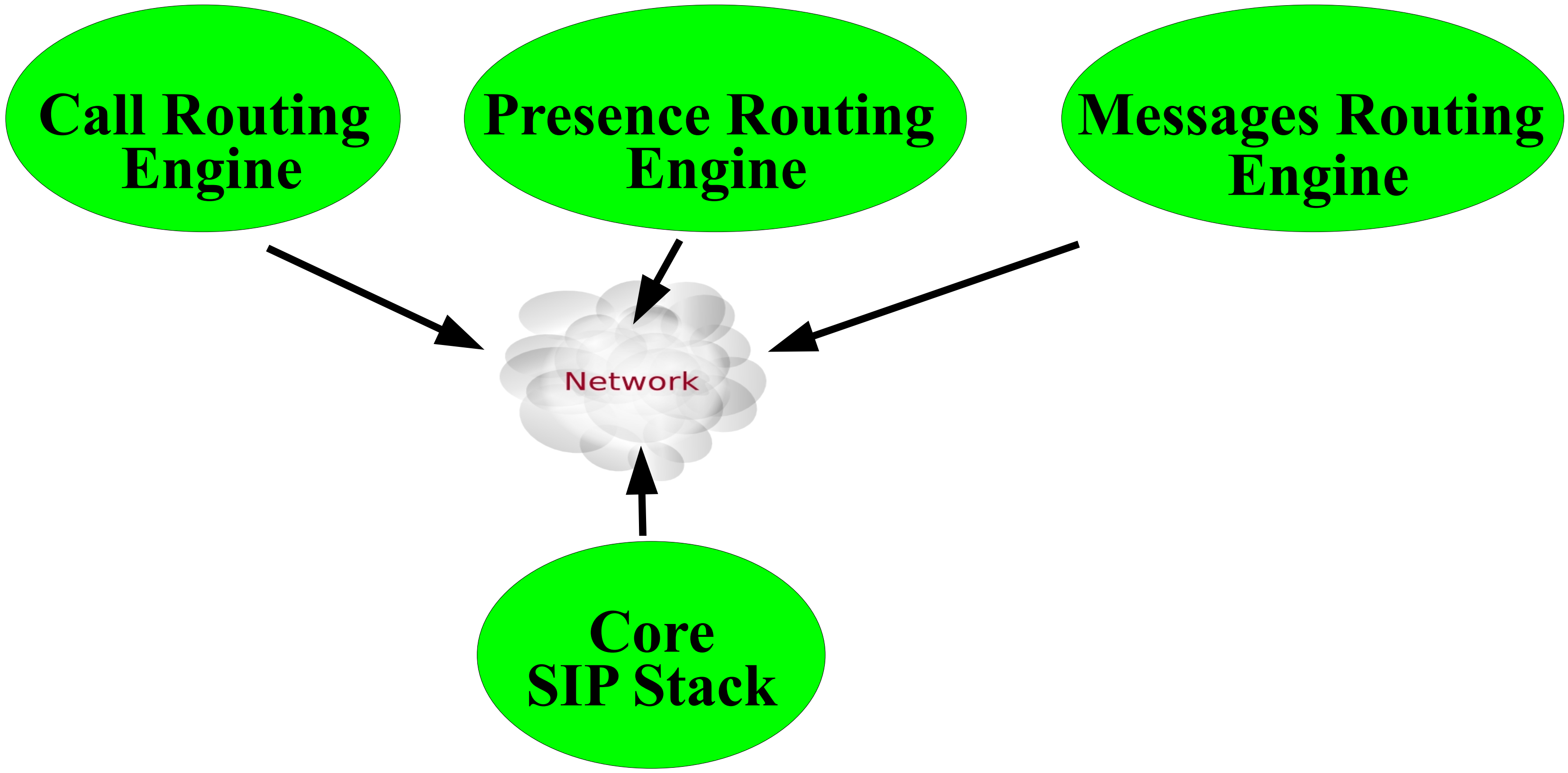
Examples

Heavy logic and integration



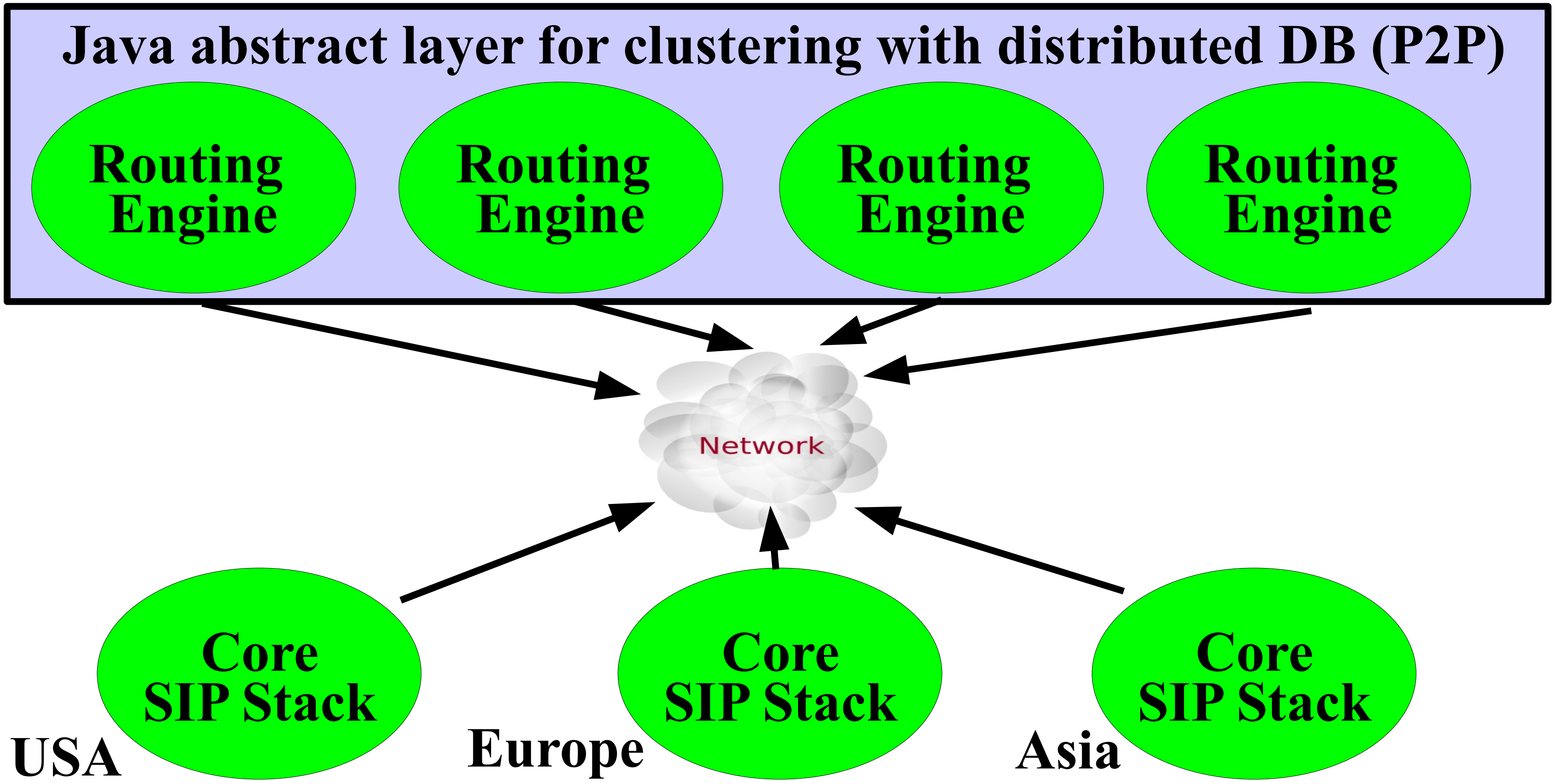
Examples

Specialized logics



Examples

Geographical Clustering



Java abstract layer for clustering with distributed DB (P2P)

Routing Engine

Routing Engine

Routing Engine

Routing Engine

Network

Core SIP Stack

Core SIP Stack

Core SIP Stack

USA

Europe

Asia

Thank you for your attention

You can find out more at www.opensips.org

bogdan@opensips.org

Questions are welcome