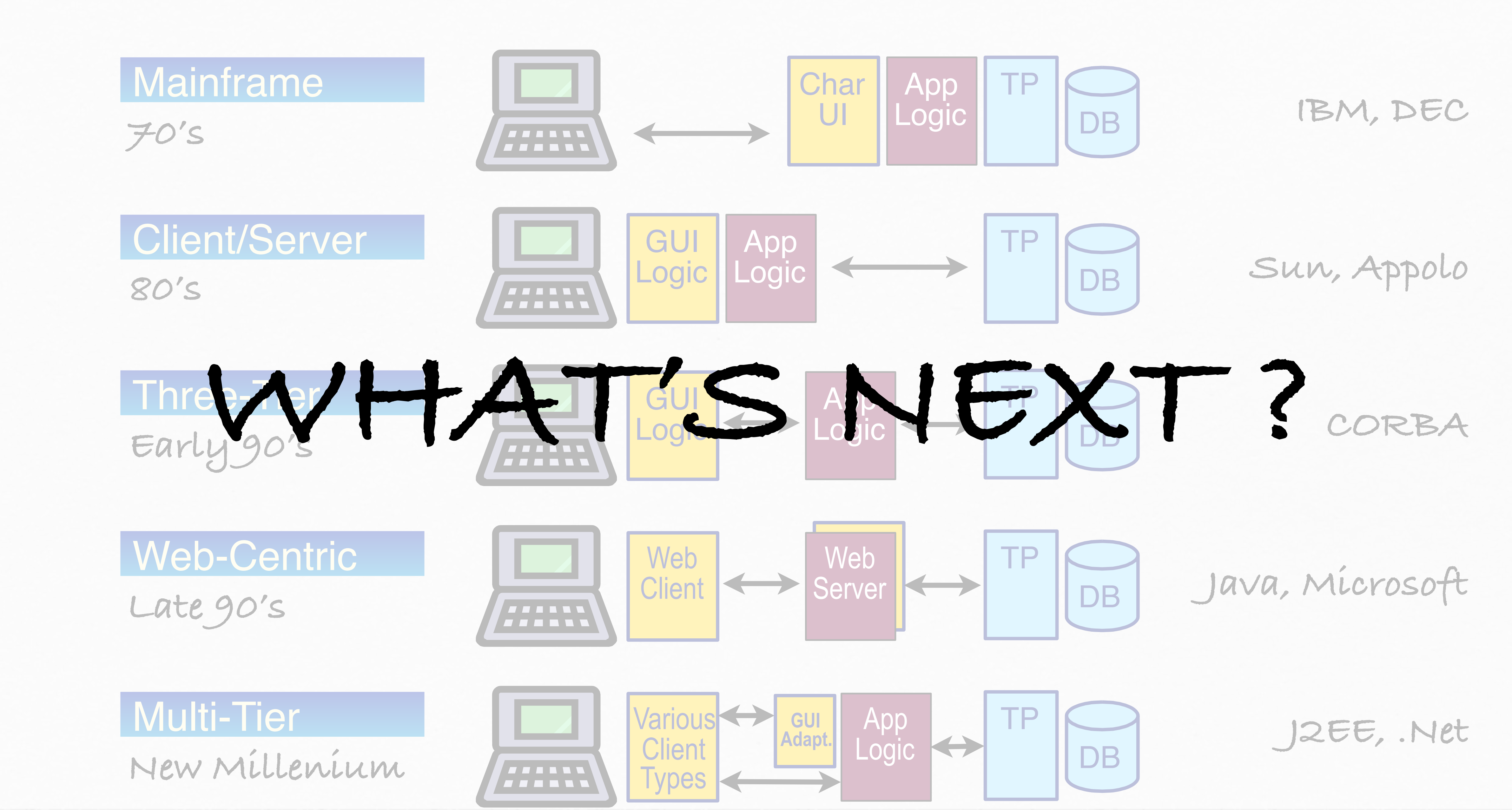
Mobile Ubiquitous Computing

Unil | HEC | dop lab

Benoît Garbinato

distributed object programming lab

Context



Mainframe

70's

Char UI

App Logic

TP

DB

IBM, DEC

Client/Server

80's

GUI Logic

App Logic

TP

DB

Sun, Appolo

Three-Tier

Early 90's

GUI Logic

App Logic

TP

DB

CORBA

Web-Centric

Late 90's

Web Client

Web Server

TP

DB

Java, Microsoft

Multi-Tier

New Millenium

Various Client Types

GUI Adapt.

App Logic

TP

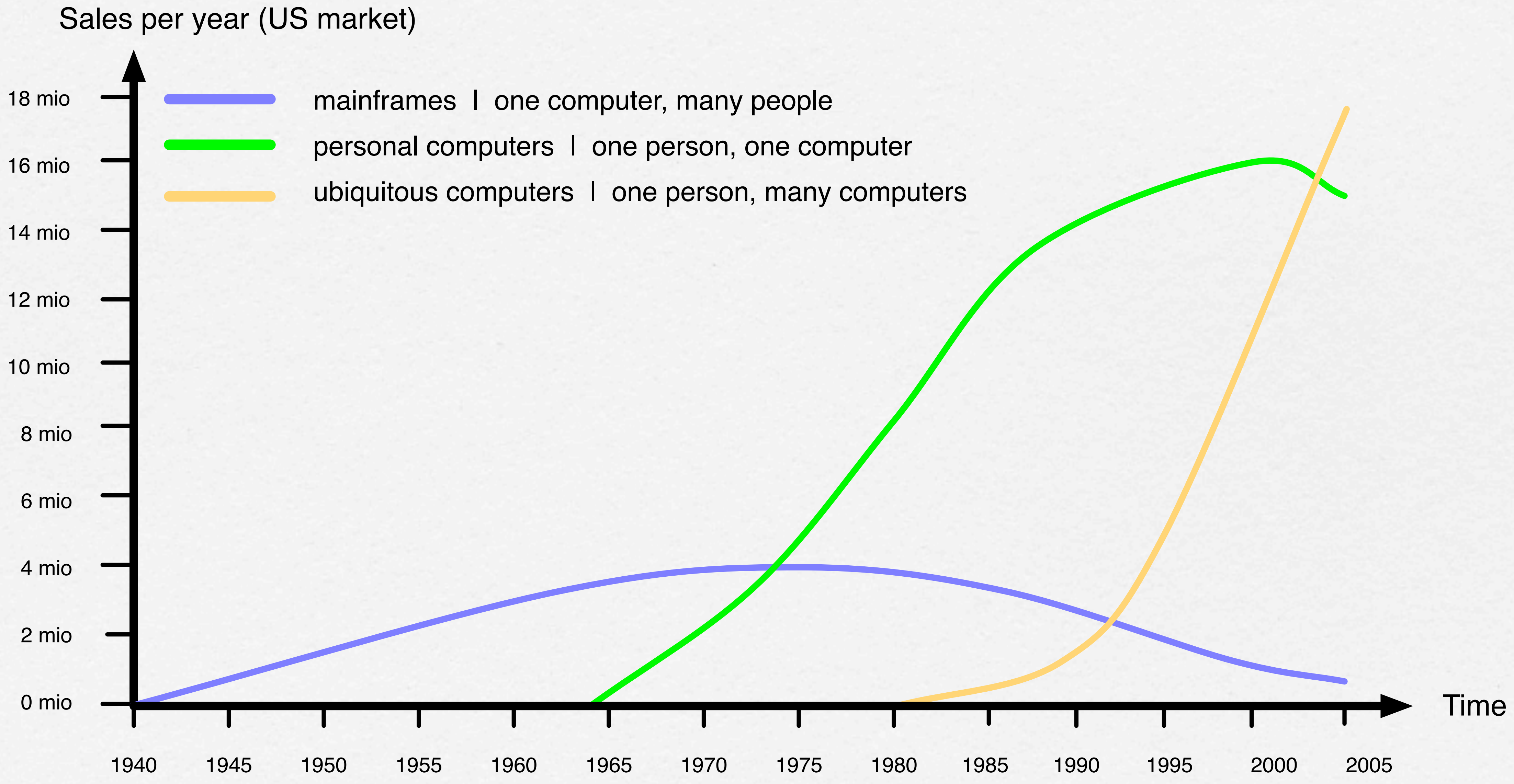
DB

J2EE, .Net

WHAT'S NEXT ?

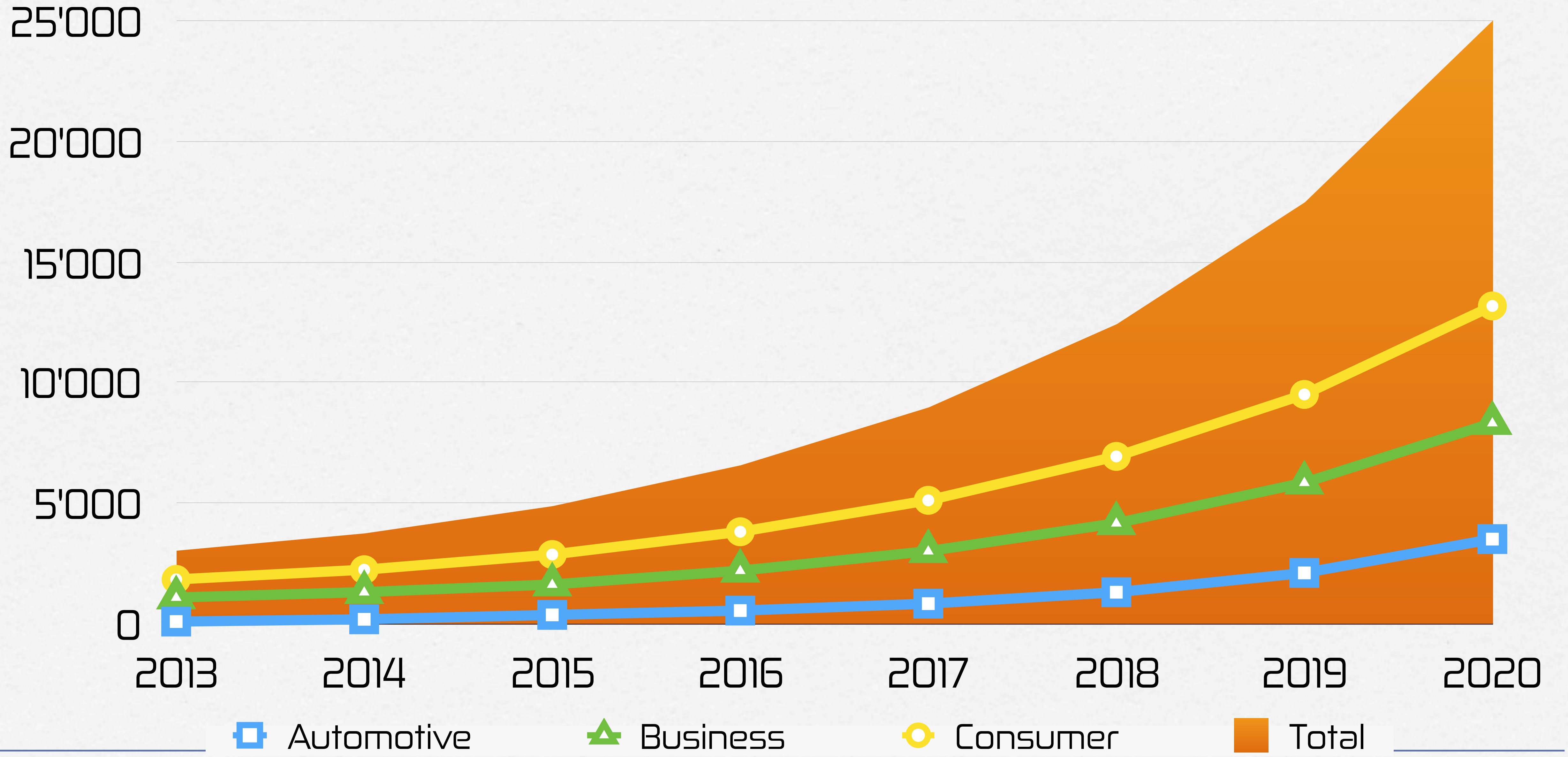
Ubiquitous computing

Yesterday...

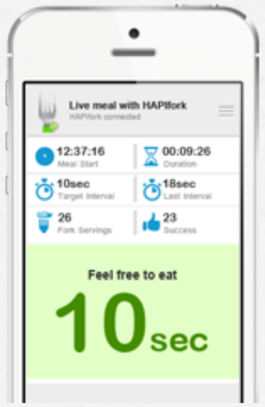


Ubiquitous computing?

Today… Internet of Things (IoT) !



Connected Objects



A Vision...

In the late nineties, Brian Halla from National Semiconductor wrote:

- □ processors will continue to become cheaper and faster,
- □ general-purpose PCs will eventually disappear,
- □ ubiquitous processors will be given for free by service providers.

Reference: Brian Halla, How the PC Will Disappear,
IEEE Computer, vol. 31, no. 12, December 98.

What definition(s)?

mobile computing

ambient intelligence

sensor networks

ubiquitous computing

context-aware computing

location-aware computing

pervasive computing

nomadic computing

mobile ad hoc networks

Mobile vs. nomadic computing

□ In common: anytime and anywhere

□ Difference:
<u>nomadic</u>: multiple fixed locations
<u>mobile</u>: continuous on-the-move operation

Context/location awareness

- □ Context awareness: the computing system is aware of its environment and acts accordingly, e.g., time, temperature, device capability, location, user interests, activity, etc.
- □ Location-awareness: a special case of context awareness (see location-based pub/sub as an example)

Ubiquitous computing

Ubiquitous computing is the method of enhancing computer use by making many computers available throughout the physical environment, but making them effectively invisible to the user.

Mark Weiser, the "father" of
ubiquitous computing

http://www.ubiq.com/hypertext/weiser/acmfuture2endnote.htm

Weiser’s vision

- Notion of “calm” technology, i.e., disappearing, invisible technology
- The computing devices is no longer at the center of our attraction, i.e., the best tools are those invisible to their users

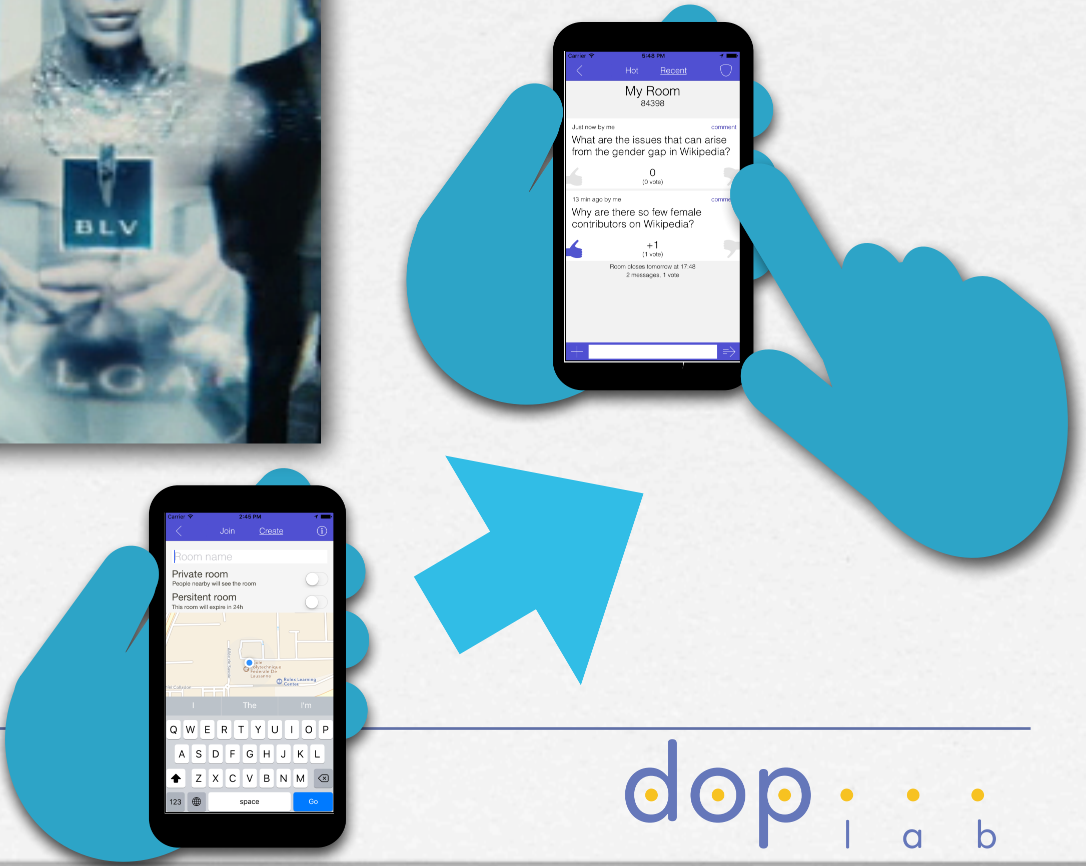
Ubiquitous computing

- □ Computing devices are immersed in an real human-based environment*
- □ Devices have limited resources, e.g., power supply, memory, bandwidth, cpu, etc.
- □ Devices are mobile and wireless, and may reside on a person (wearable computing)

*somehow the dual of virtual reality, where humans are immersed in a virtual computer-based environment

Some scenarios…

R.C. Central Park
50 Central Park South, New York
The R.C. New York Central Park is a luxurious haven in the heart of the city.



My Room
84398

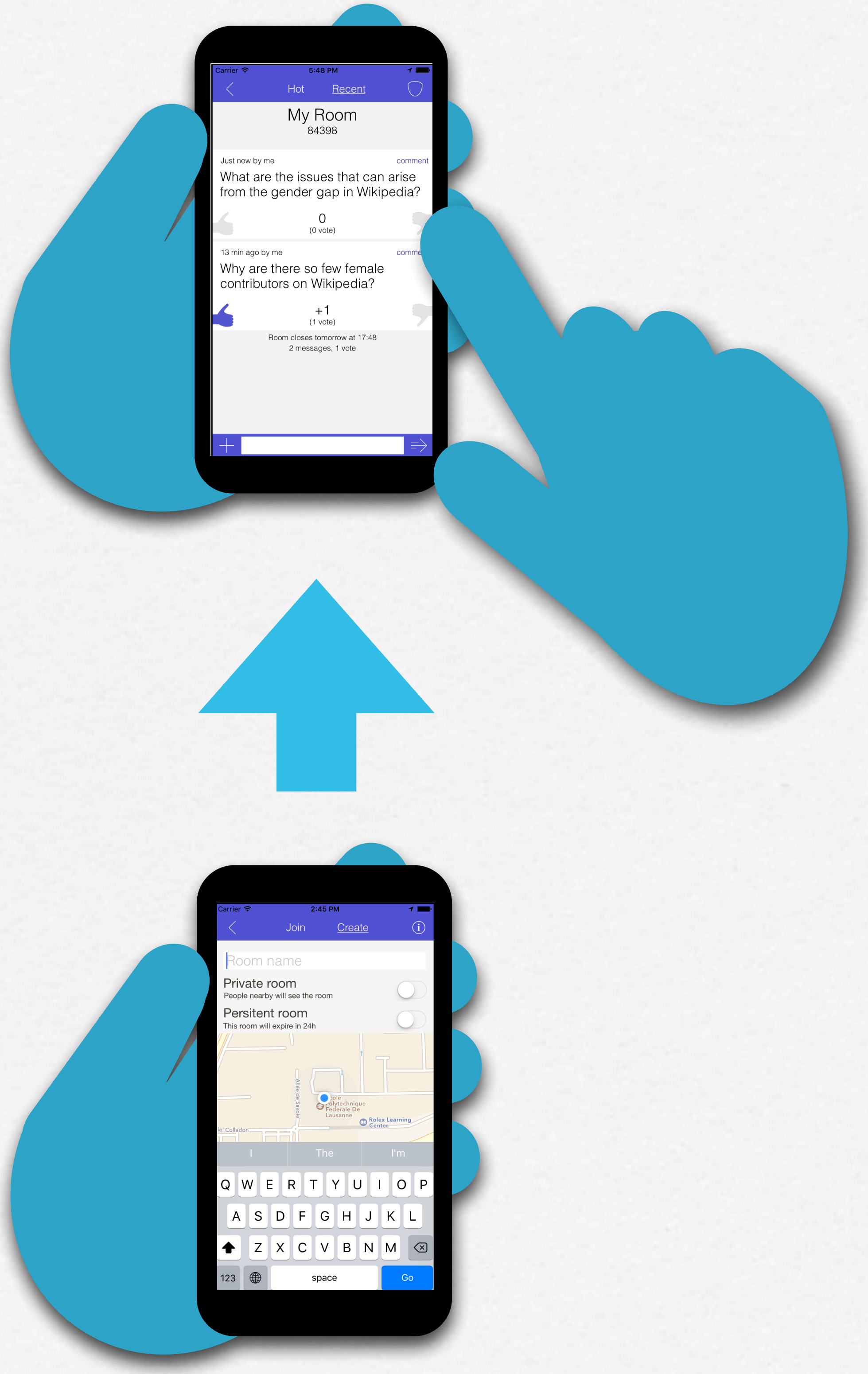
Just now by me
What are the issues that can arise from the gender gap in Wikipedia?
0
(0 vote)

13 min ago by me
Why are there so few female contributors on Wikipedia?
+1
(1 vote)

Room closes tomorrow at 17:48
2 messages, 1 vote

Room name
Private room ○
People nearby will see the room
Persitent room ○
This room will expire in 24h

SpeakUp



My Room
84398

Just now by me
comment

What are the issues that can arise from the gender gap in Wikipedia?

0
(0 vote)

13 min ago by me
comme

Why are there so few female contributors on Wikipedia?

+1
(1 vote)

Room closes tomorrow at 17:48
2 messages, 1 vote

Room name

Private room
People nearby will see the room

Persitent room
This room will expire in 24h

http://speakup.info

What devices?

- □ PDA and smart phones
- □ Smart devices/cards, e.g., Java card, iButtons, etc.
- □ Radio Frequency ID tags (RFIDs)
- □ Sensors networks

- □ Embedded systems, e.g., in the automotive industry
- □ ...

Distributed computing issues

- Remote communication (RMI, MOM, etc.)
- Naming of distributed entities/services
- Distributed data management, e.g., distributed file systems, distributed transactions, etc.
- Reliability, availability, security
- Caching (for performance)

Mobile computing issues

- □ Networking: mobile IP address, TCP de-/re-connection, performance, etc.
- □ Information access (bandwidth)
- □ Power consumption (variable cpu/disk speed, network de-/re-connection,etc.)
- □ Location awareness and resource discovery
- □ Mobile ad hoc networks & topology control

Ubiquitous computing issues

- □ Constraints on sensor design (size, cost, power consumption, etc.)
- □ Mobile ad hoc networks & topology control
- □ localized scalability (greater distance ⇒ less communication)
- □ Invisibility (millions of sensors should not distract the user)

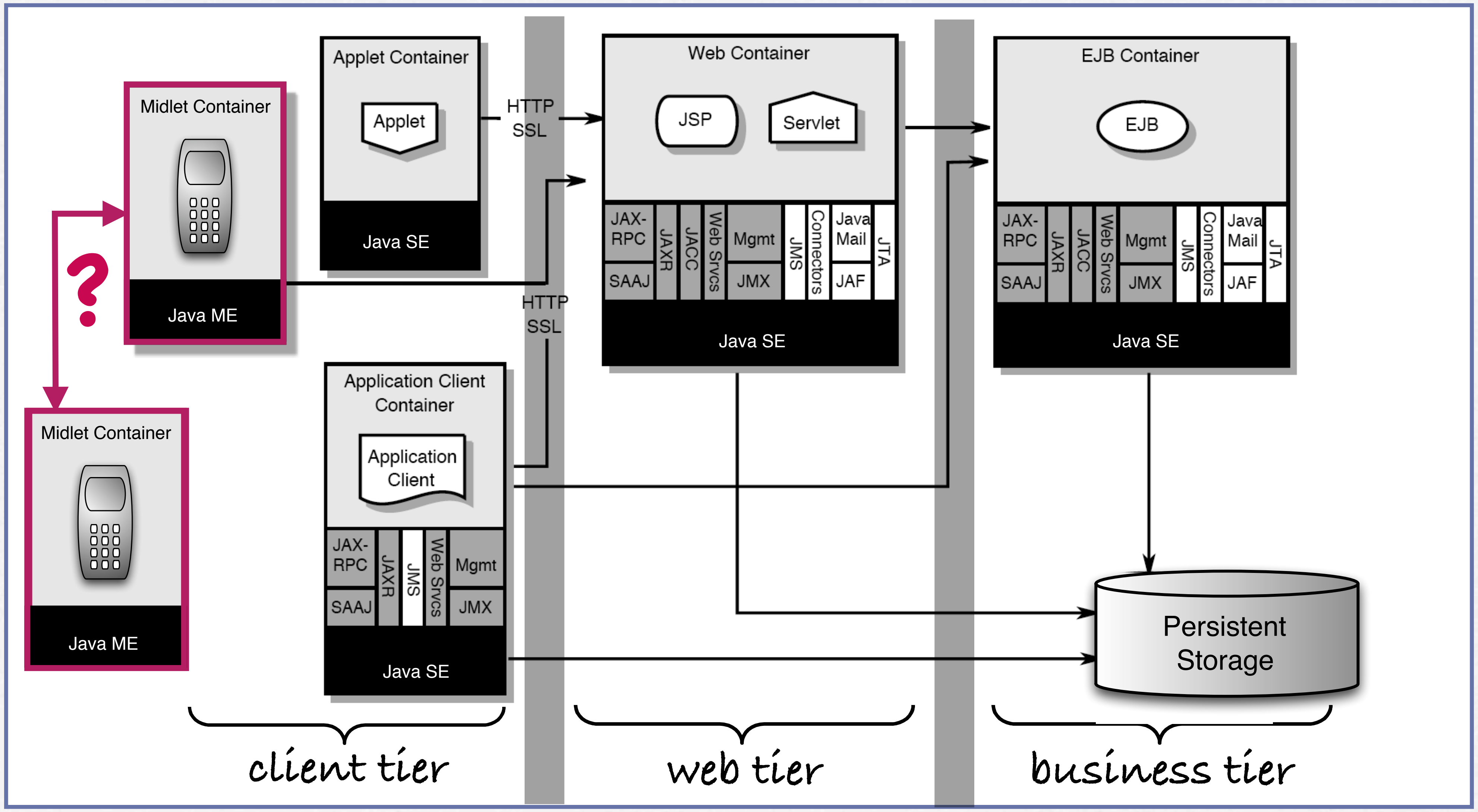
What middleware support?

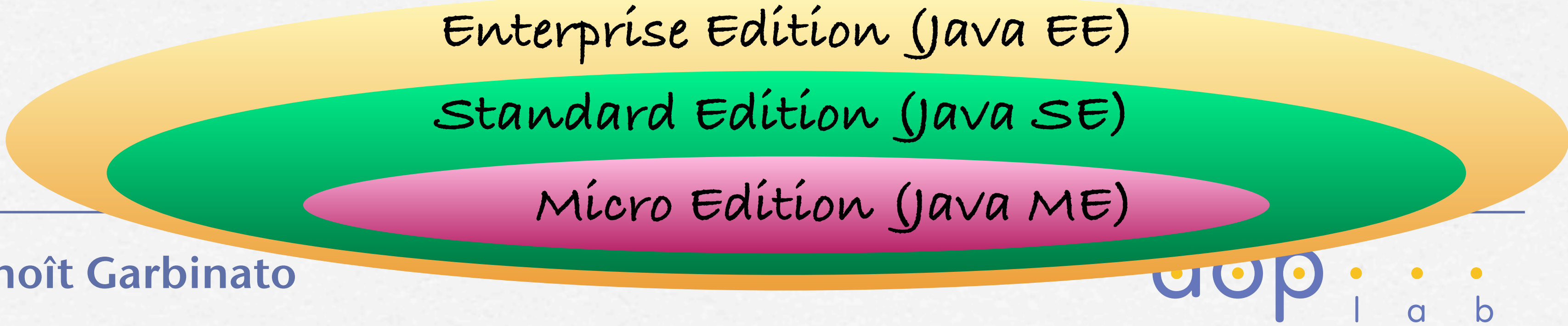
- The main challenge for a middleware supporting mobile & ubiquitous computing lies in the heterogeneity of devices
- There exist(ed) several industrial middleware:
  - Microsoft .NET Compact Framework (NETCF)
  - Qualcomm's Binary Runtime Environment for Wireless (BREW)
  - Sun Java Micro Edition (Java ME)
  - Android platform
  - Apple platform

BREW & NETCF

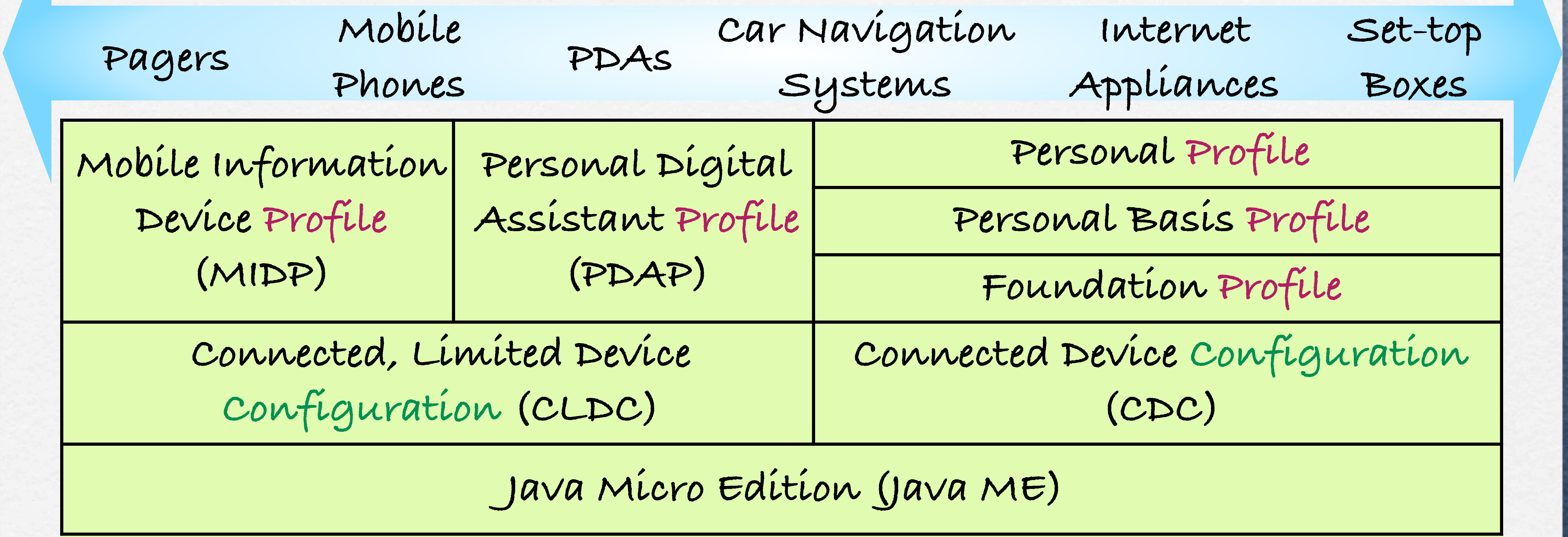
- About Qualcomm's BREW:
  - it is both an application execution environment based on C++ and a business model for operator revenue,
  - it also support Java, via a JVM built on top of it.
- About Microsoft's NETCF:
  - it is the the latest initiative from Microsoft to compete with Java ME and BREW,
  - lacks market penetration, due to a small number of devices using Windows as operating system.

The Java ME platform





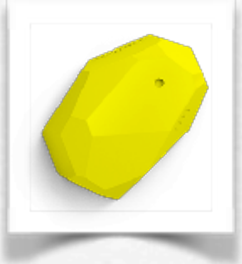
Java Micro Edition (JavaME)


Pagers
Mobile Phones
PDAs
Car Navigation Systems
Internet Appliances
Set-top Boxes
Mobile Information Device Profile (MIDP)
Personal Digital Assistant Profile (PDAP)
Personal Profile
Personal Basis Profile
Foundation Profile
Connected, Limited Device Configuration (CLDC)
Connected Device Configuration (CDC)
Java Micro Edition (Java ME)

Based on two key concepts:

- □ Java ME configurations
- □ Java ME profiles

Android platform



- □ Based on the acquisition of a small startup by Google in 2005
- □ In 2007, the Open Handset Alliance was funded to drive the development of open standards for mobile devices
- □ In 2008, Android became an open source project

- □ The development framework is based on the Java programming languages but not on standard Java APIs (neither Java SE nor Java ME)
- □ This is not (yet?) a curated platform

Apple Platform

- □ Based on Mac OS
- □ Based on Objective-C frameworks, now Swift
- □ Integrated in XCode, together with an emulator and a set of deployment options
- □ Comes with a business & application provisioning model, "à la" iTunes Store
- □ This is a curated platform, with an innovative revenue sharing models for developers

Open challenges

facing an api jungle when developing

Open challenges

facing an api jungle when developing

especially

no programming
support combining

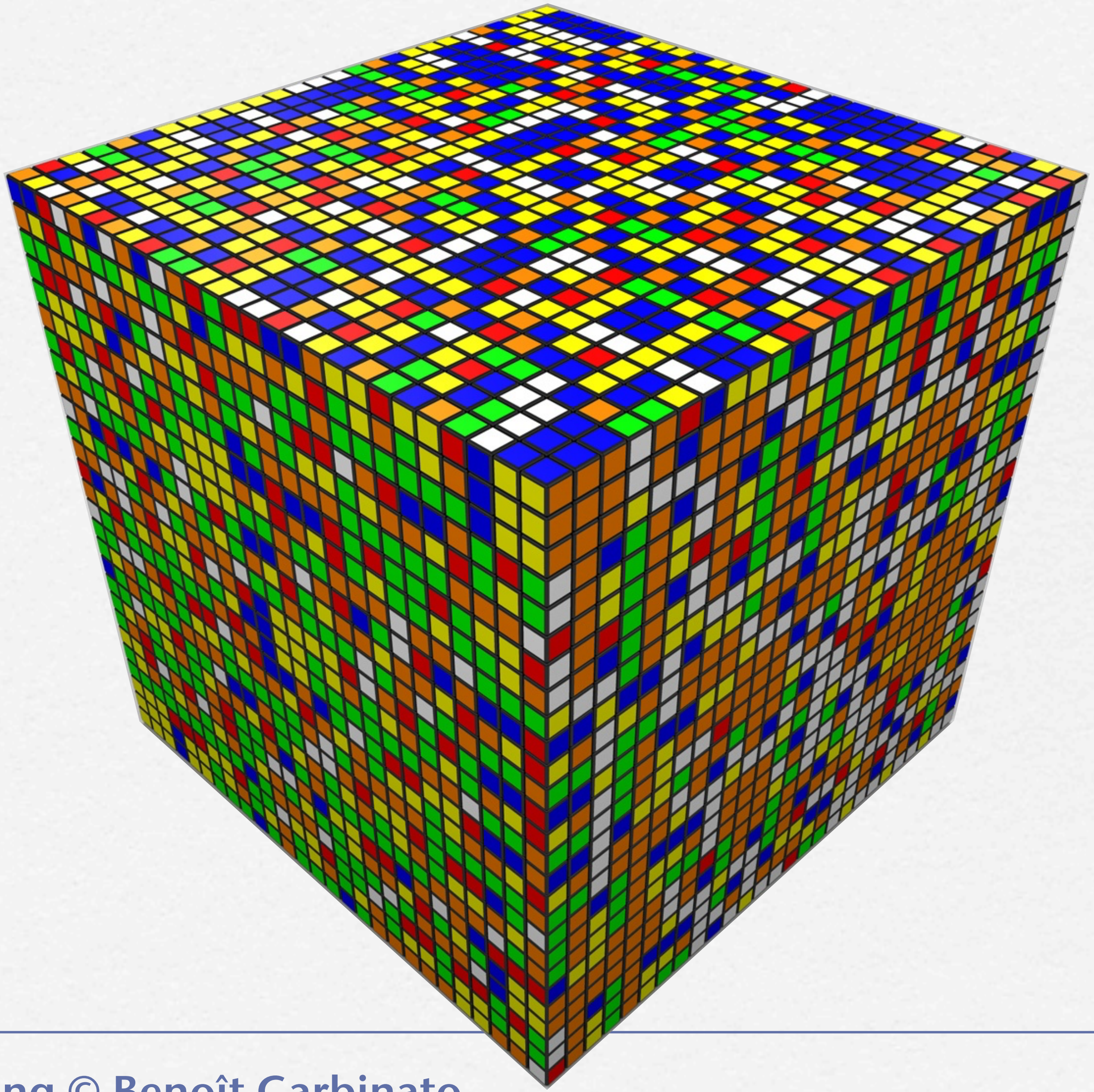
communication

+

sensory input

Open challenges

facing a scalability wall when deploying



Open Challenges

each connected object can be seen as a moving producer and consumer of contextual information that needs to be tracked

mobile app developers face complex development and deployment issues even for simple context-aware services

development

multiple hardware, operating systems, protocols, etc.

⬇

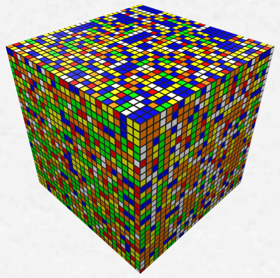
the api jungle challenge

deployment

massive tracking, messaging, testing, monitoring, etc.

⬇

the scalability challenge



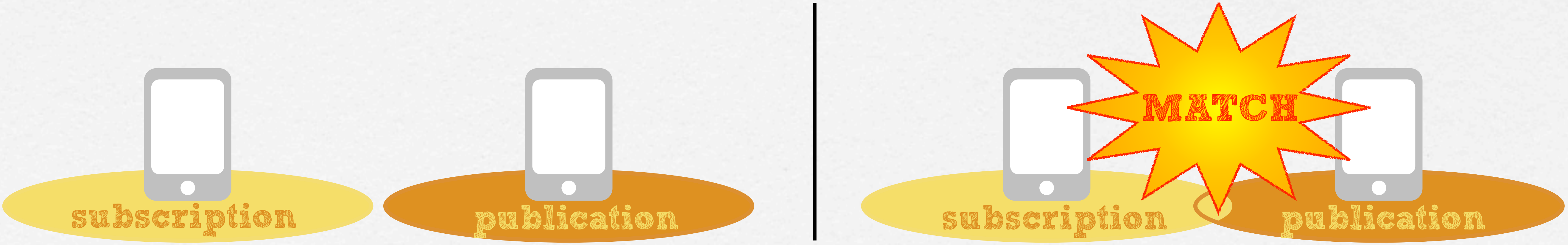
Publish/Subscribe

anonymous

asynchronous

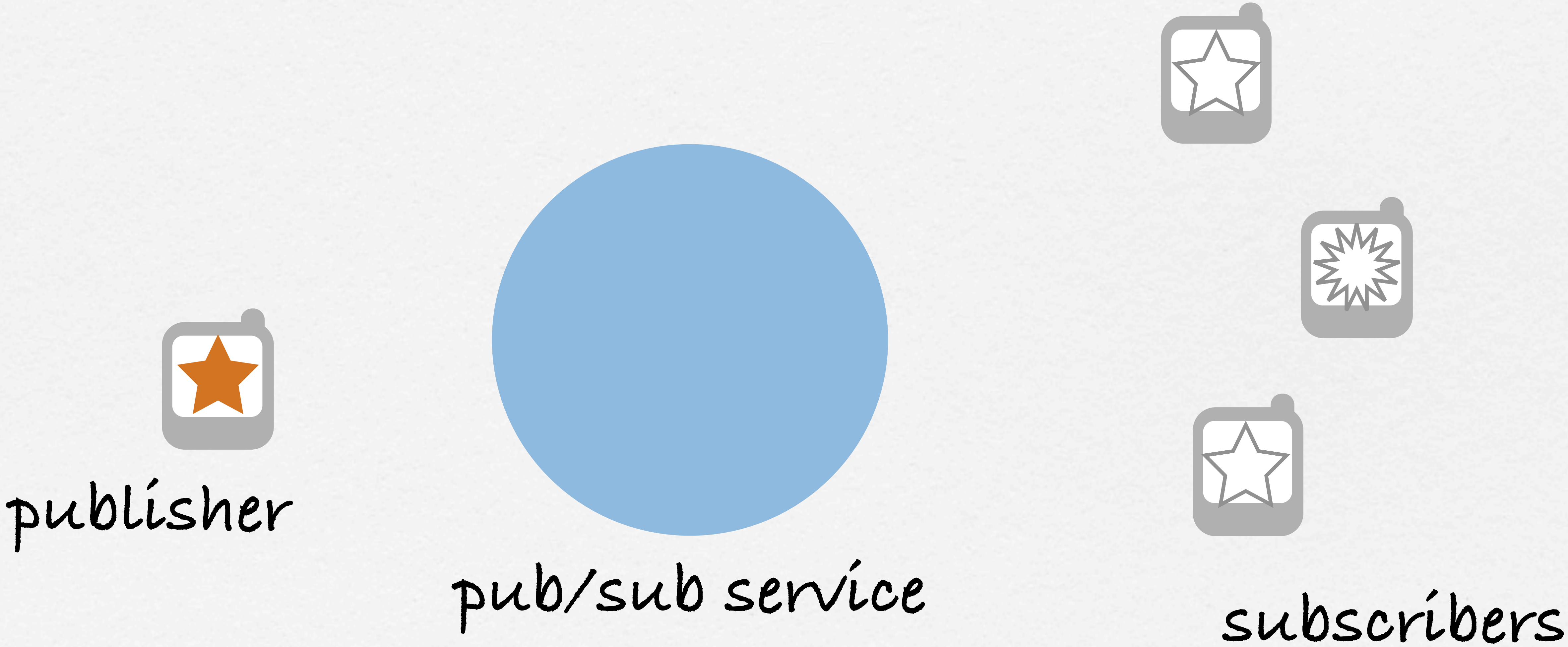
+

Context Awareness



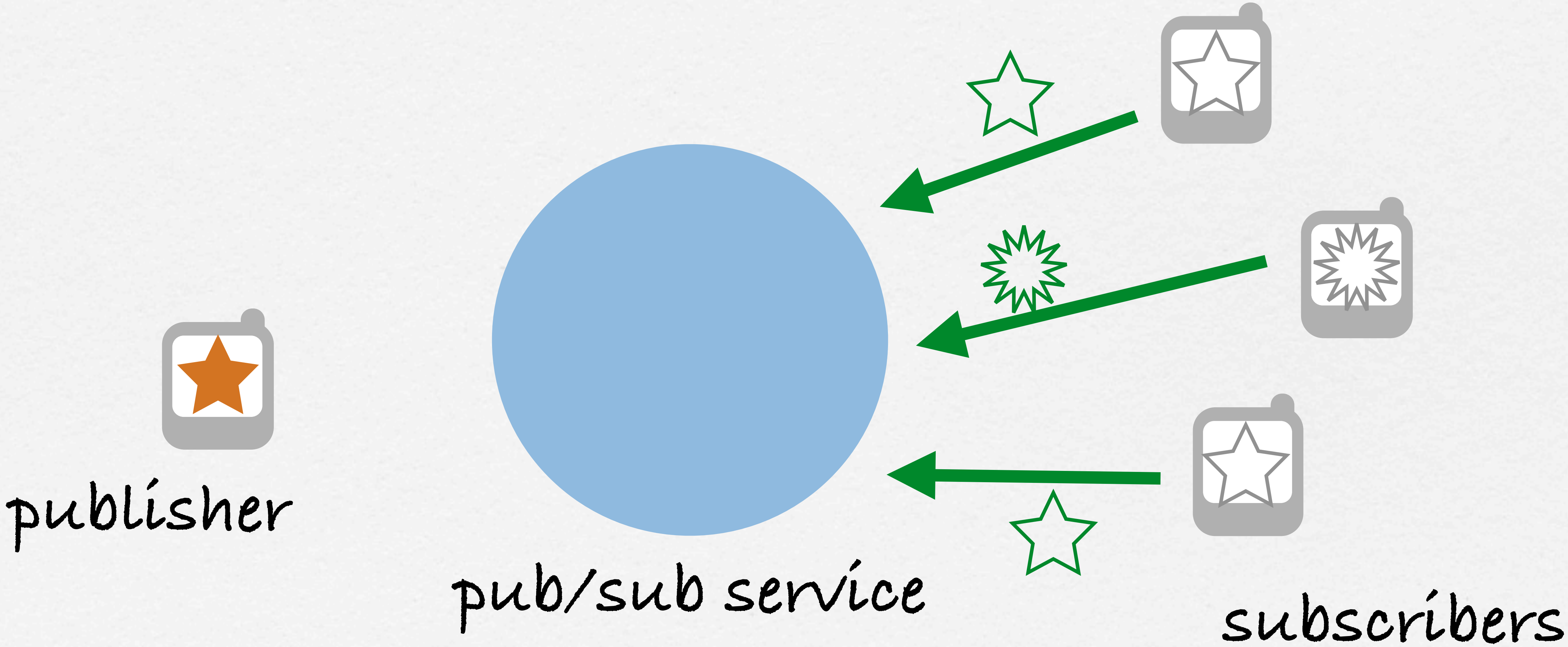
Publish/Subscribe

as starting point



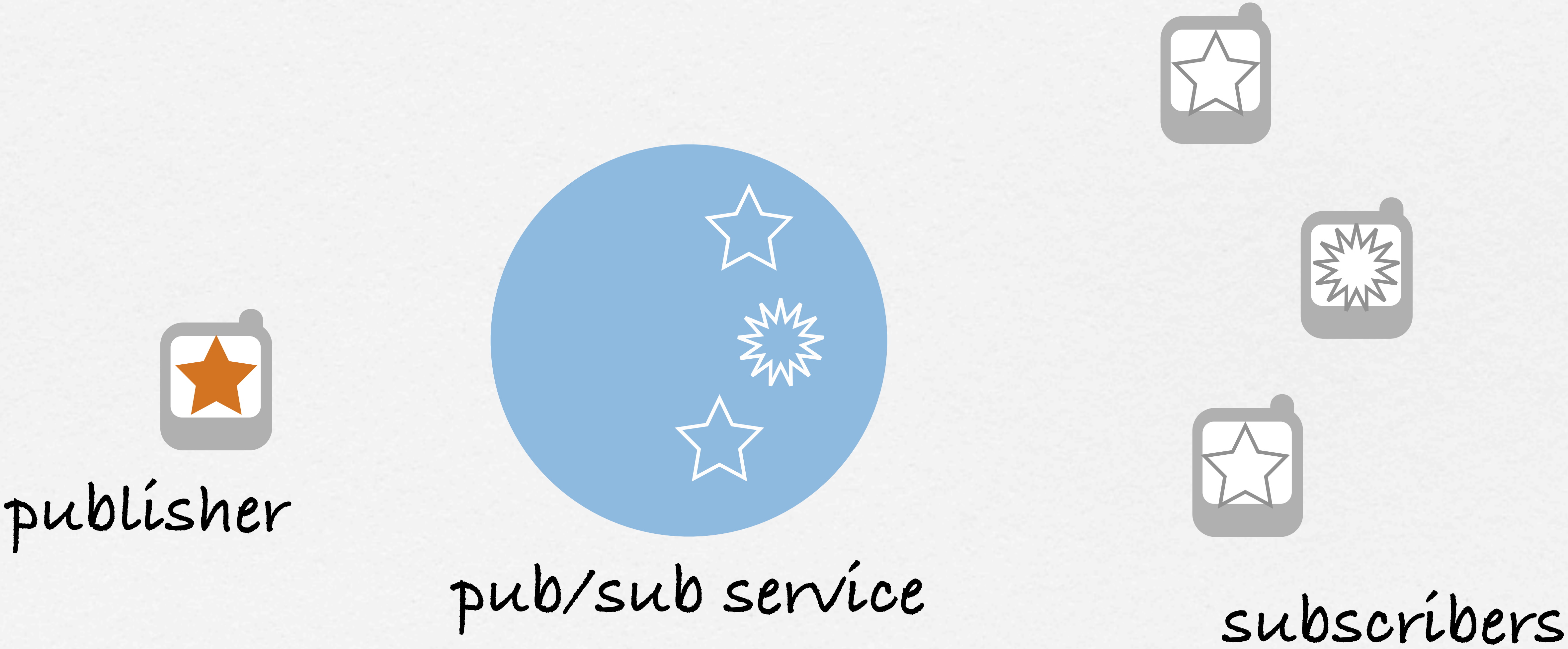
Publish/Subscribe

subscriptions are created



Publish/Subscribe

subscriptions are created



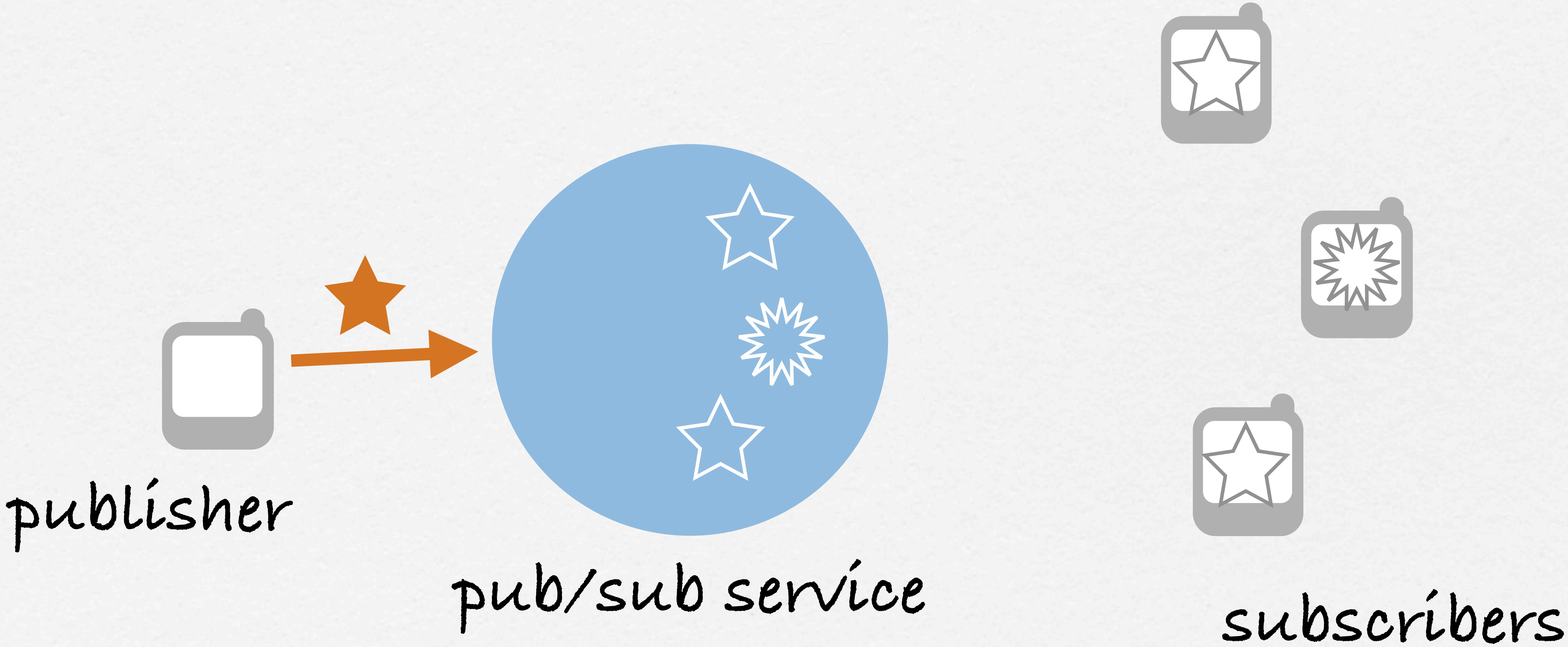
publisher

pub/sub service

subscribers

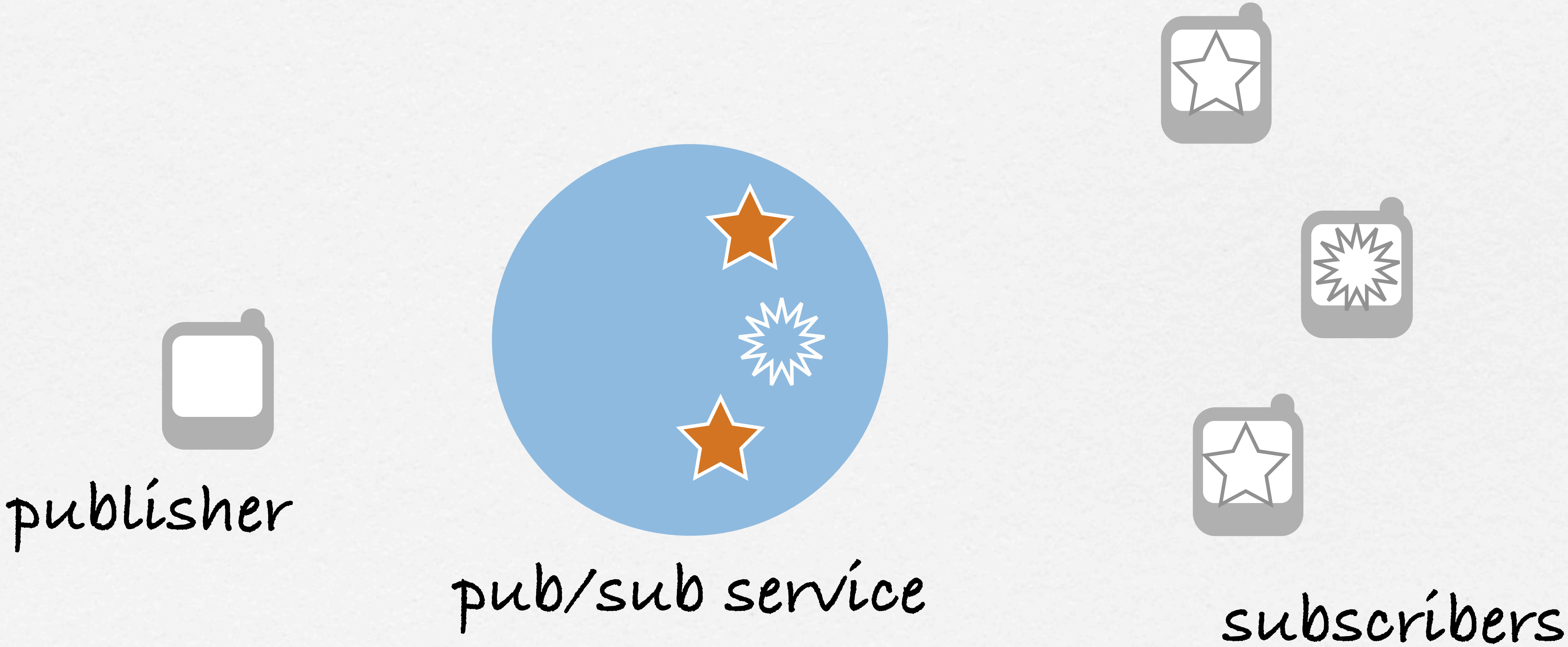
Publish/Subscribe

message is published



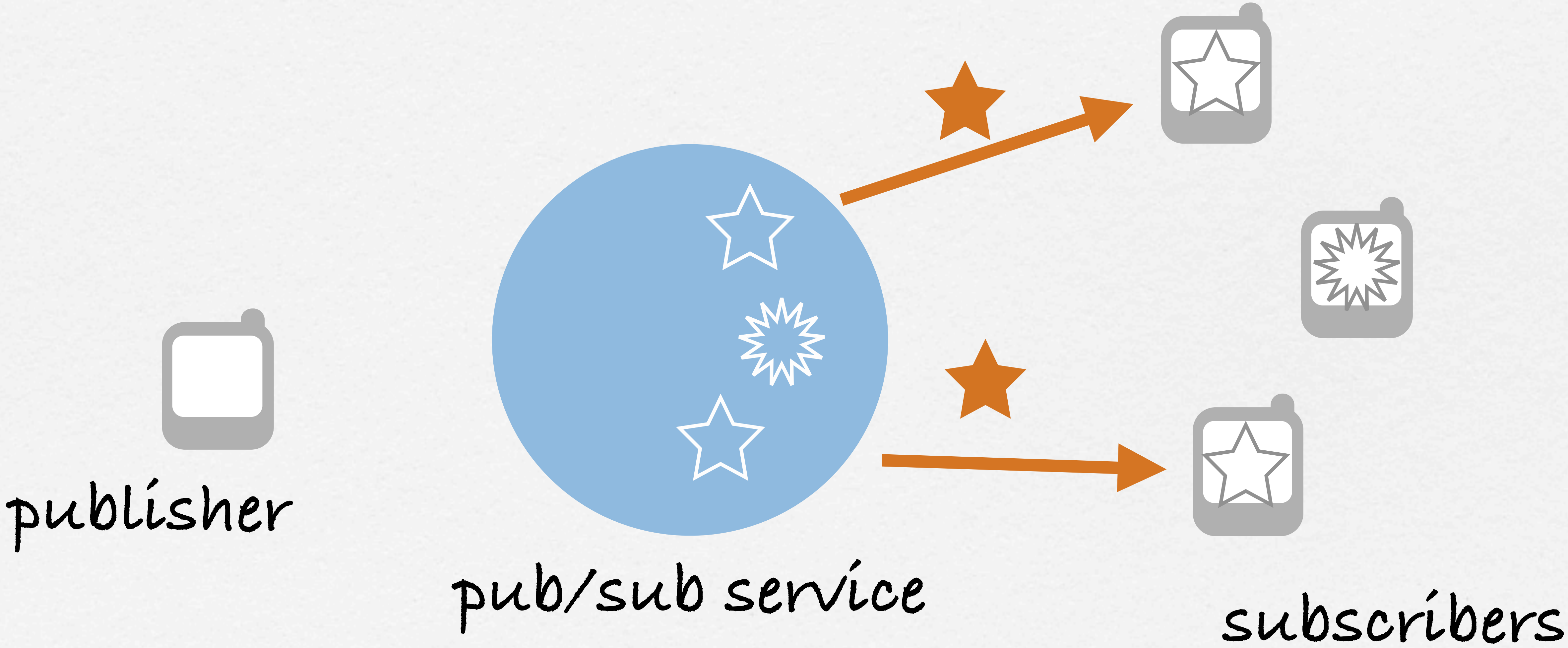
Publish/Subscribe

content match occurs



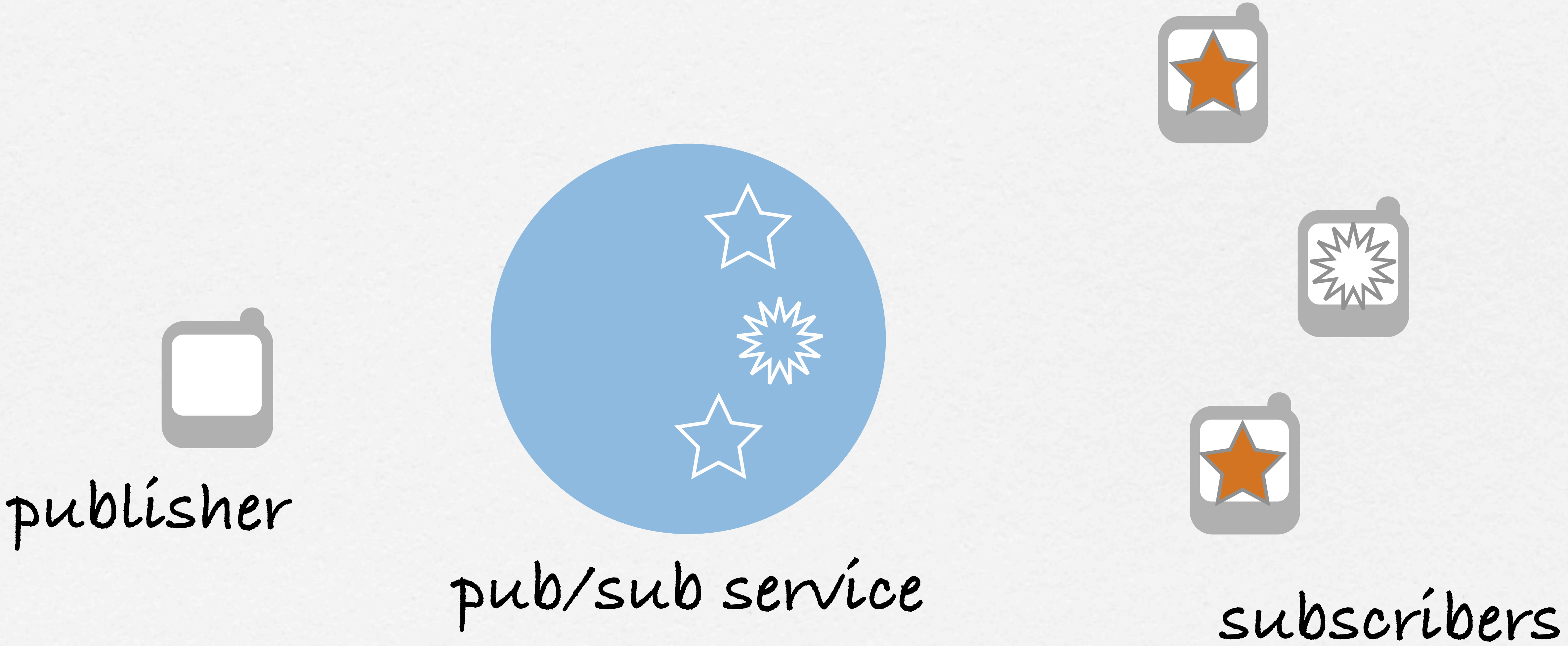
Publish/Subscribe

message is delivered

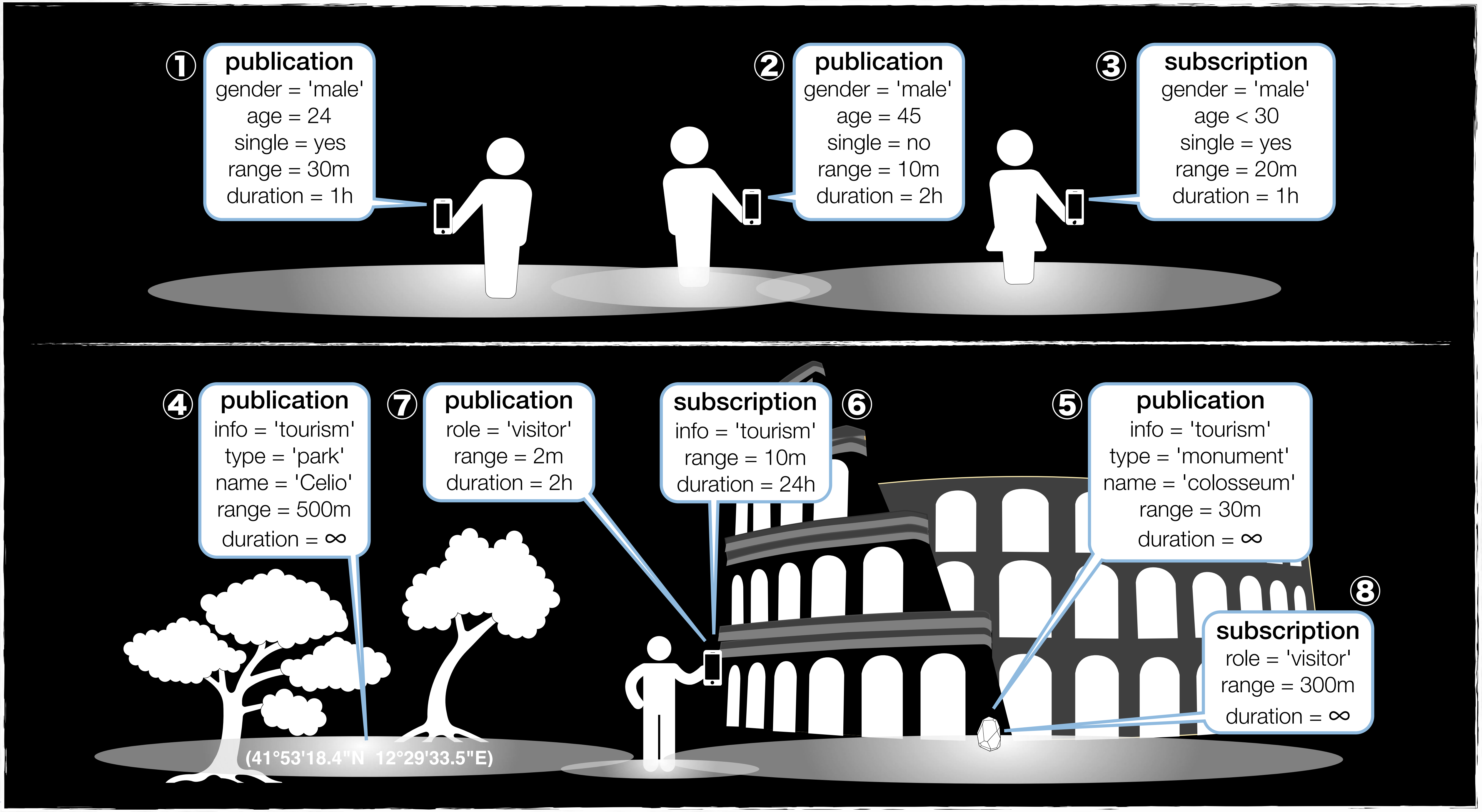


Publish/Subscribe

message is delivered



Location-based Publish/Subscribe



① publication
gender = 'male'
age = 24
single = yes
range = 30m
duration = 1h

② publication
gender = 'male'
age = 45
single = no
range = 10m
duration = 2h

③ subscription
gender = 'male'
age < 30
single = yes
range = 20m
duration = 1h

④ publication
info = 'tourism'
type = 'park'
name = 'Celio'
range = 500m
duration = ∞

⑦ publication
role = 'visitor'
range = 2m
duration = 2h

⑥ subscription
info = 'tourism'
range = 10m
duration = 24h

⑤ publication
info = 'tourism'
type = 'monument'
name = 'colosseum'
range = 30m
duration = ∞

⑧ subscription
role = 'visitor'
range = 300m
duration = ∞

Location-based Publish/Subscribe

the idea is to make it possible to build

UBER™

tinder™

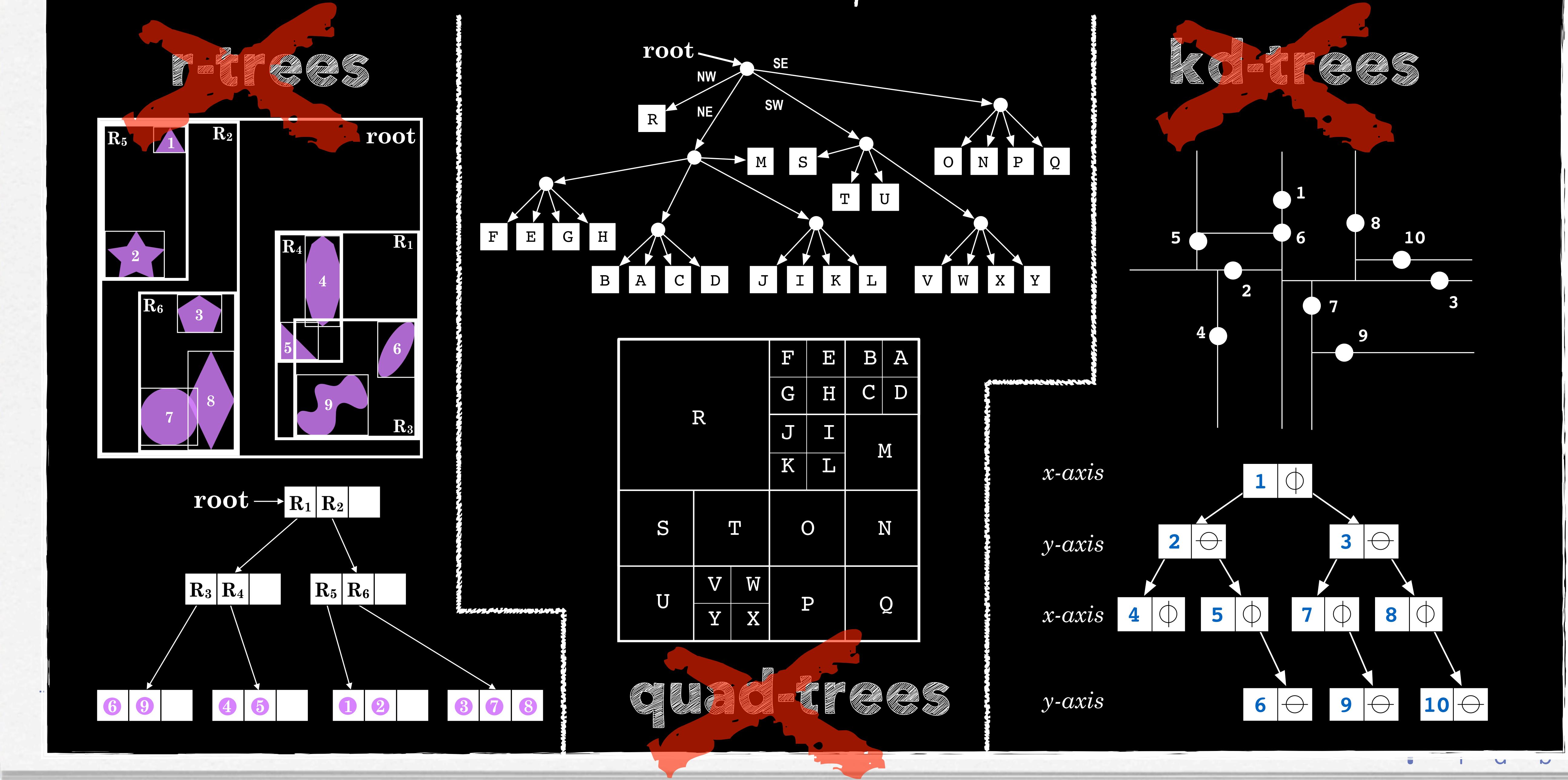
Pokémon GO™

in 30 minutes

Location-based Publish/Subscribe

Algorithmic Challenge

existing spatial indexing structures are totally inefficient when both read and write operations are intensive


r-trees
R5
1
R2
root
2
R4
R1
4
R6
3
5
6
8
7
9
R3
root
R1 R2
R3 R4
R5 R6
6 9
4 5
1 2
3 7 8
root
NW
SE
NE
SW
R
M
S
O N P Q
T U
F E G H
B A C D
J I K L
V W X Y
F E B A
G H C D
J I
K L
M
S T O N
U
V W
Y X
P Q
quad-trees
kd-trees
1
5
6
8
10
2
3
7
4
9
x-axis
y-axis
x-axis
y-axis
1
2
3
4
5
7
8
6
9
10