

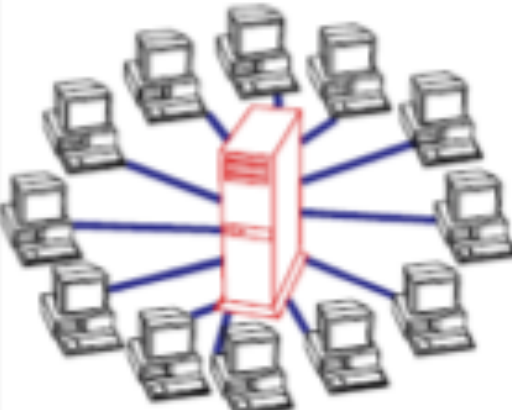



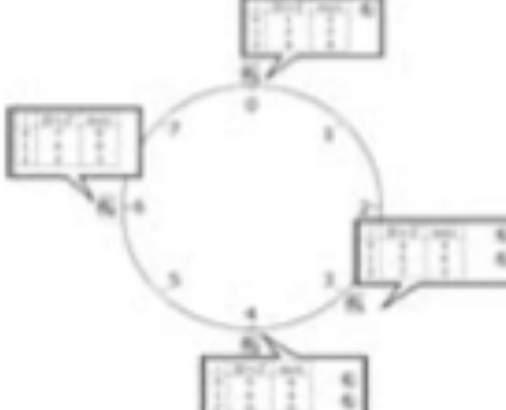
# Client-Server vs Peer to Peer

Which is better?

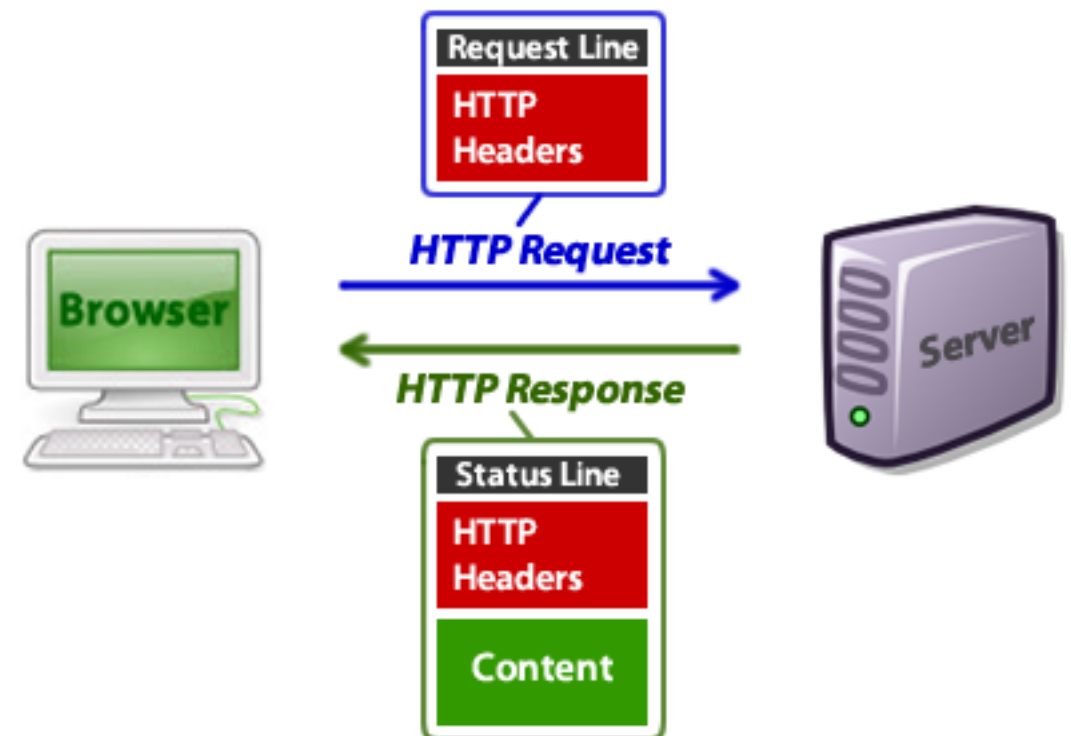
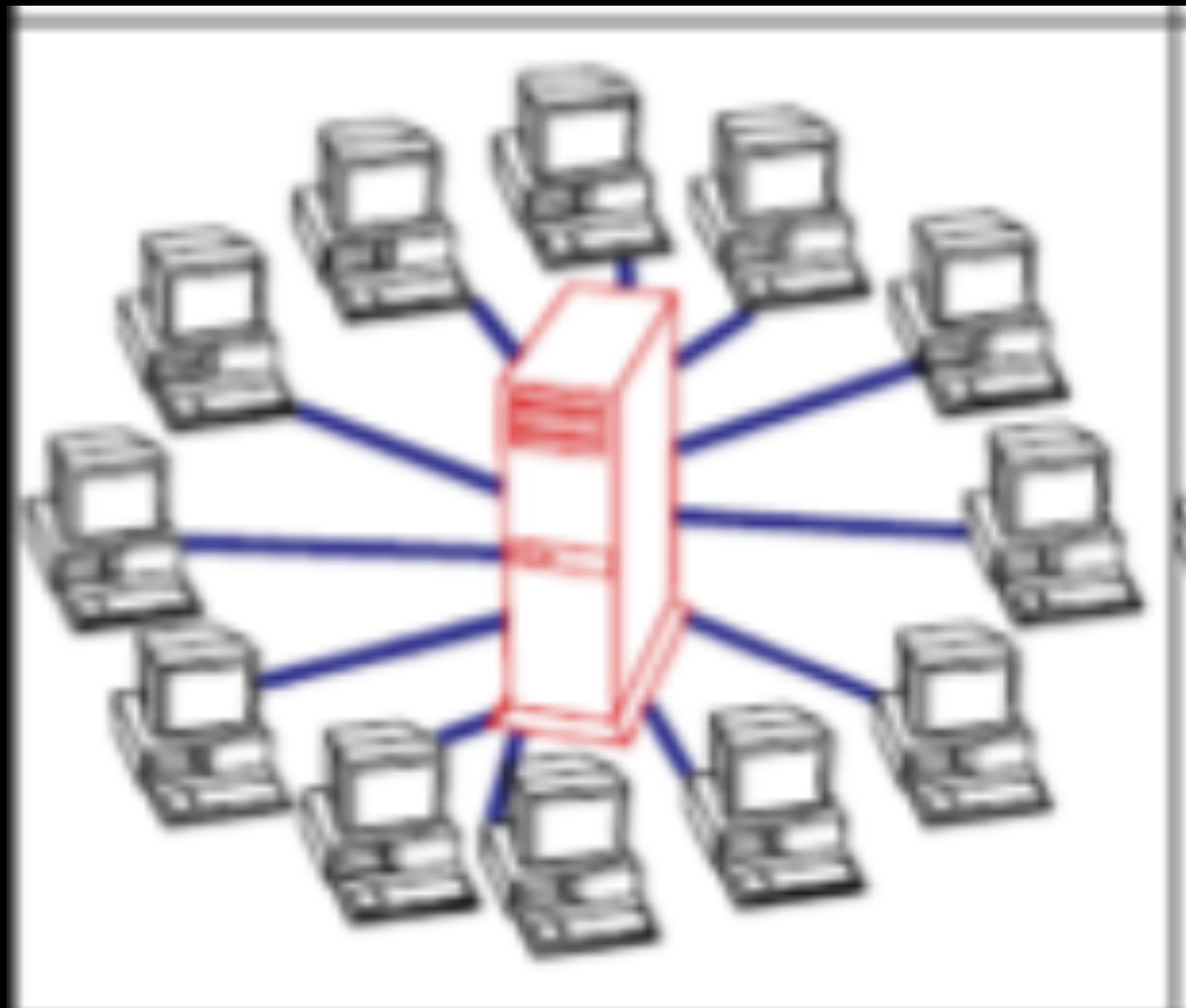
NEITHER!

# Importance in Ubicomp

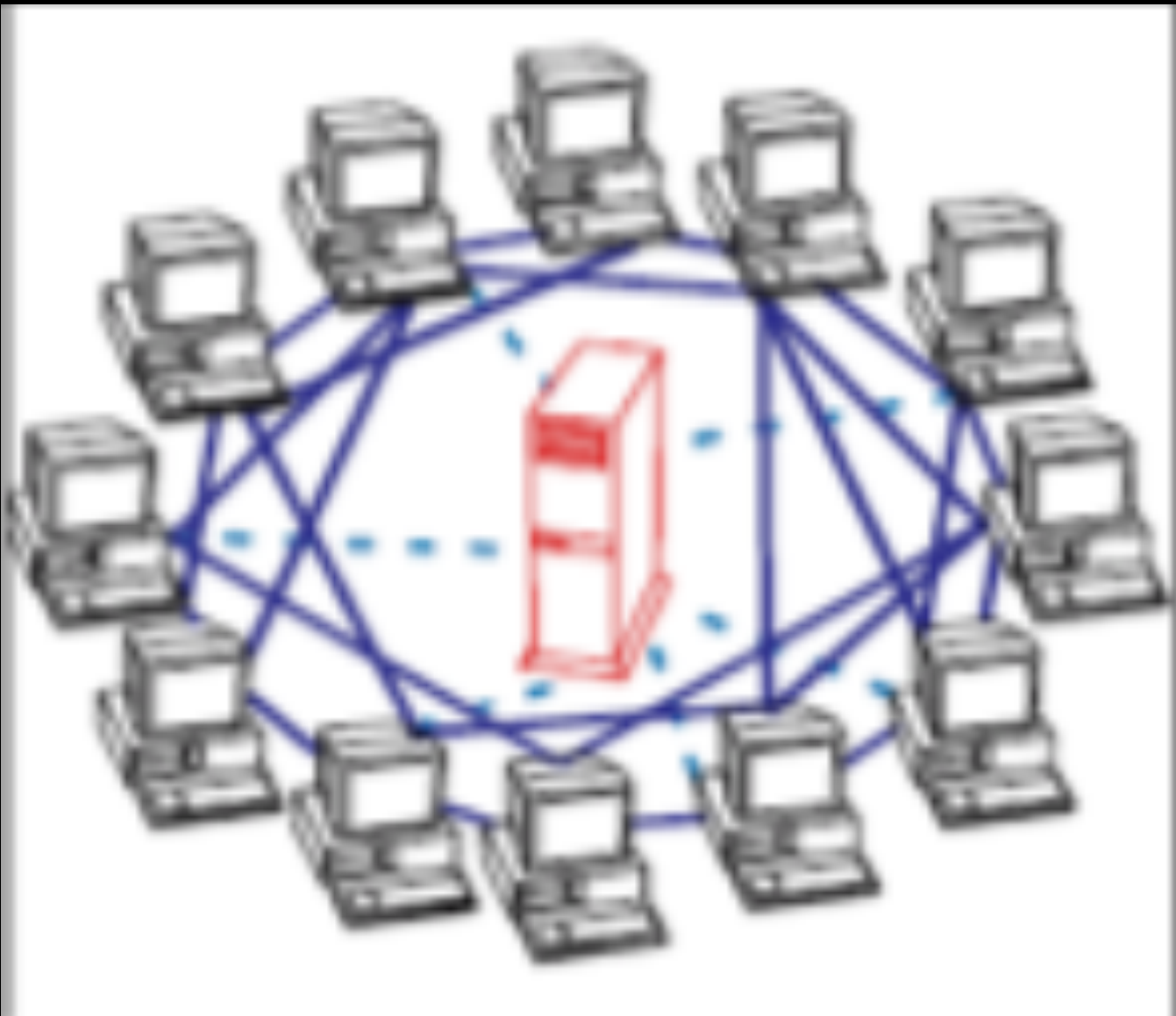
- More devices = more data
- More communication
- More storage
- More processing

<i><b>Client-Server</b></i>	<i><b>Peer-to-Peer</b></i>			
	<ol style="list-style-type: none"> <li>Resources are shared between the peers</li> <li>Resources can be accessed directly from other peers</li> <li>Peer is provider and requestor (Servent concept)</li> </ol>			
	<i><b>Unstructured P2P</b></i>			<i><b>Structured P2P</b></i>
	<i><b>1st Generation</b></i>		<i><b>2nd Generation</b></i>	
<ol style="list-style-type: none"> <li>Server is the central entity and only provider of service and content. → Network managed by the Server</li> <li>Server as the higher performance system.</li> <li>Clients as the lower performance system</li> </ol> <p>Example: WWW</p>	<i><b>Centralized P2P</b></i>	<i><b>Pure P2P</b></i>	<i><b>Hybrid P2P</b></i>	<i><b>DHT-Based</b></i>
	<ol style="list-style-type: none"> <li>All features of Peer-to-Peer included</li> <li>Central entity is necessary to provide the service</li> <li>Central entity is some kind of index/group database</li> </ol> <p>Example: Napster</p>	<ol style="list-style-type: none"> <li>All features of Peer-to-Peer included</li> <li>Any terminal entity can be removed without loss of functionality</li> <li>→ No central entities</li> </ol> <p>Examples: Gnutella 0.4, Freenet</p>	<ol style="list-style-type: none"> <li>All features of Peer-to-Peer included</li> <li>Any terminal entity can be removed without loss of functionality</li> <li>→ dynamic central entities</li> </ol> <p>Example: Gnutella 0.6, JXTA</p>	<ol style="list-style-type: none"> <li>All features of Peer-to-Peer included</li> <li>Any terminal entity can be removed without loss of functionality</li> <li>→ No central entities</li> <li>Connections in the overlay are "fixed"</li> </ol> <p>Examples: Chord, CAN</p>
				

# Client-Server

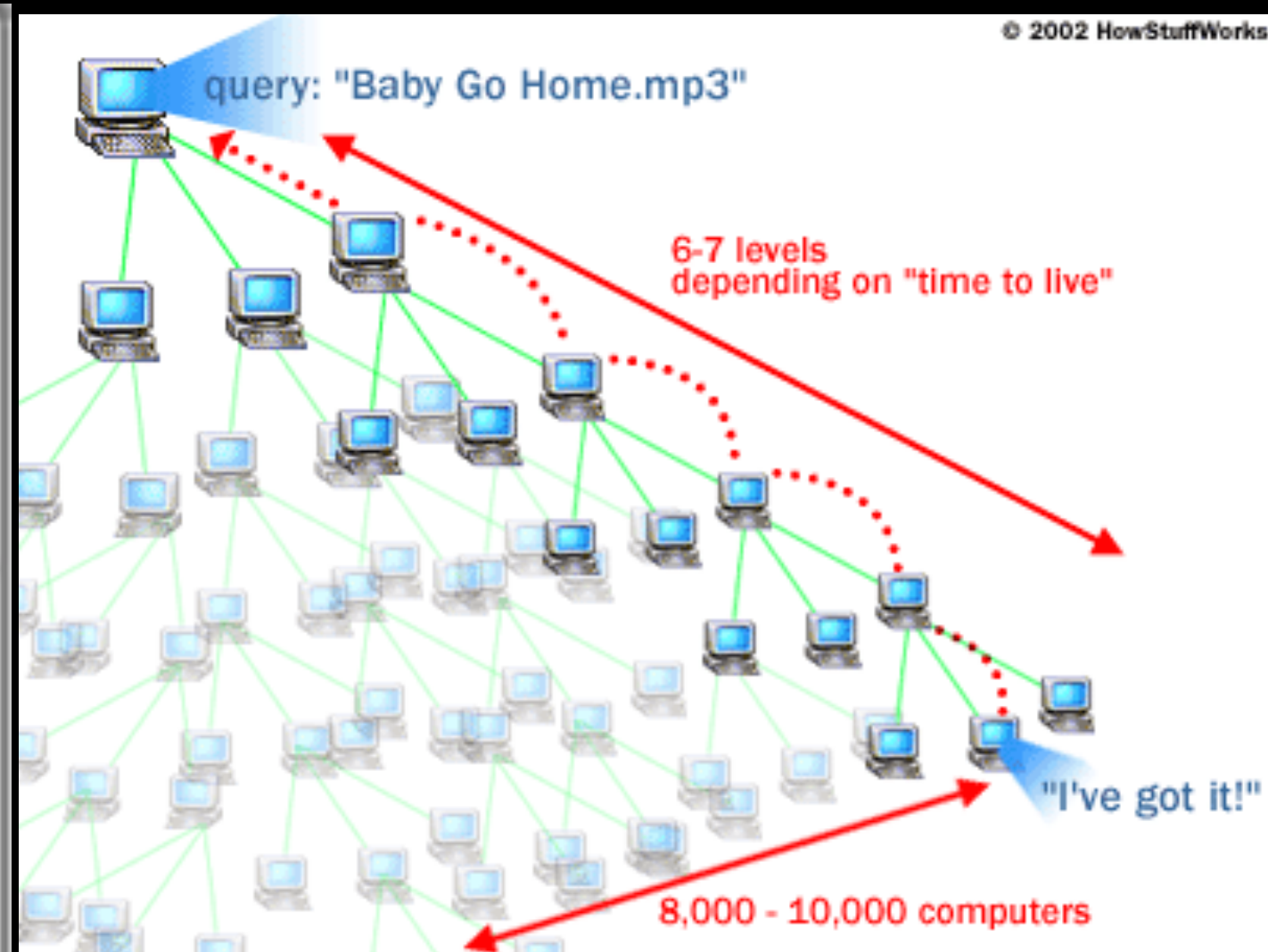
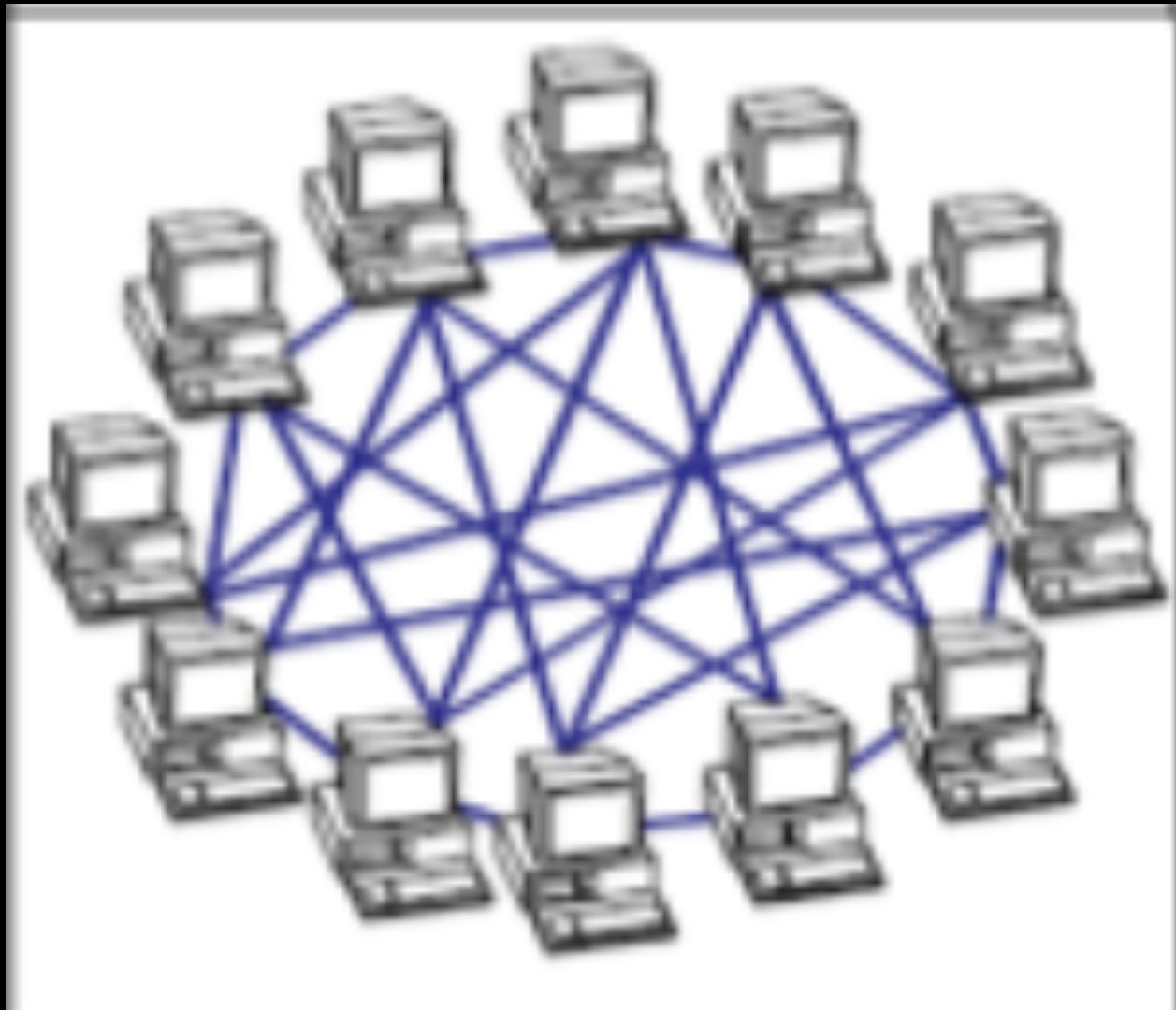


# Centralised Peer-to-Peer

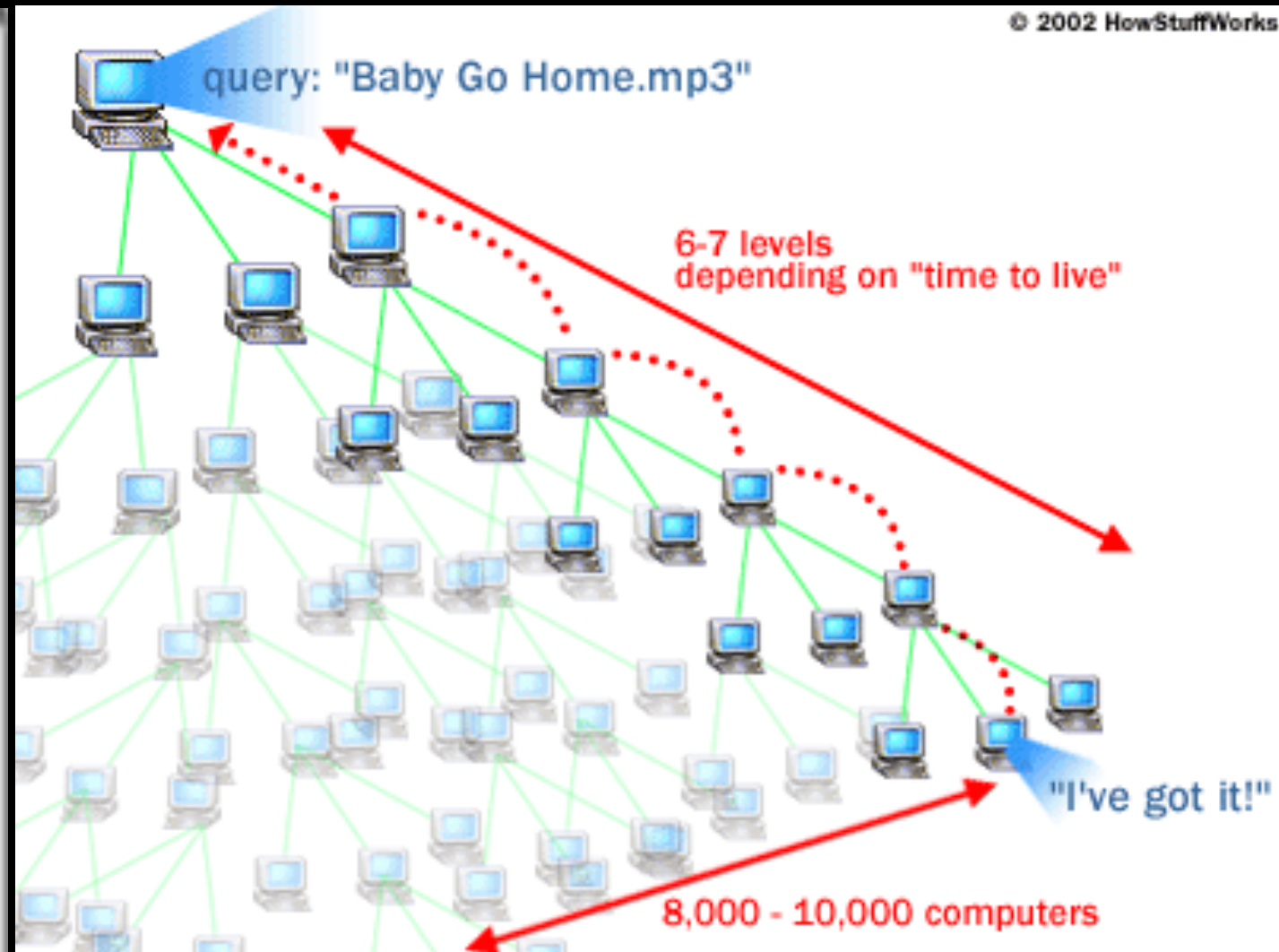
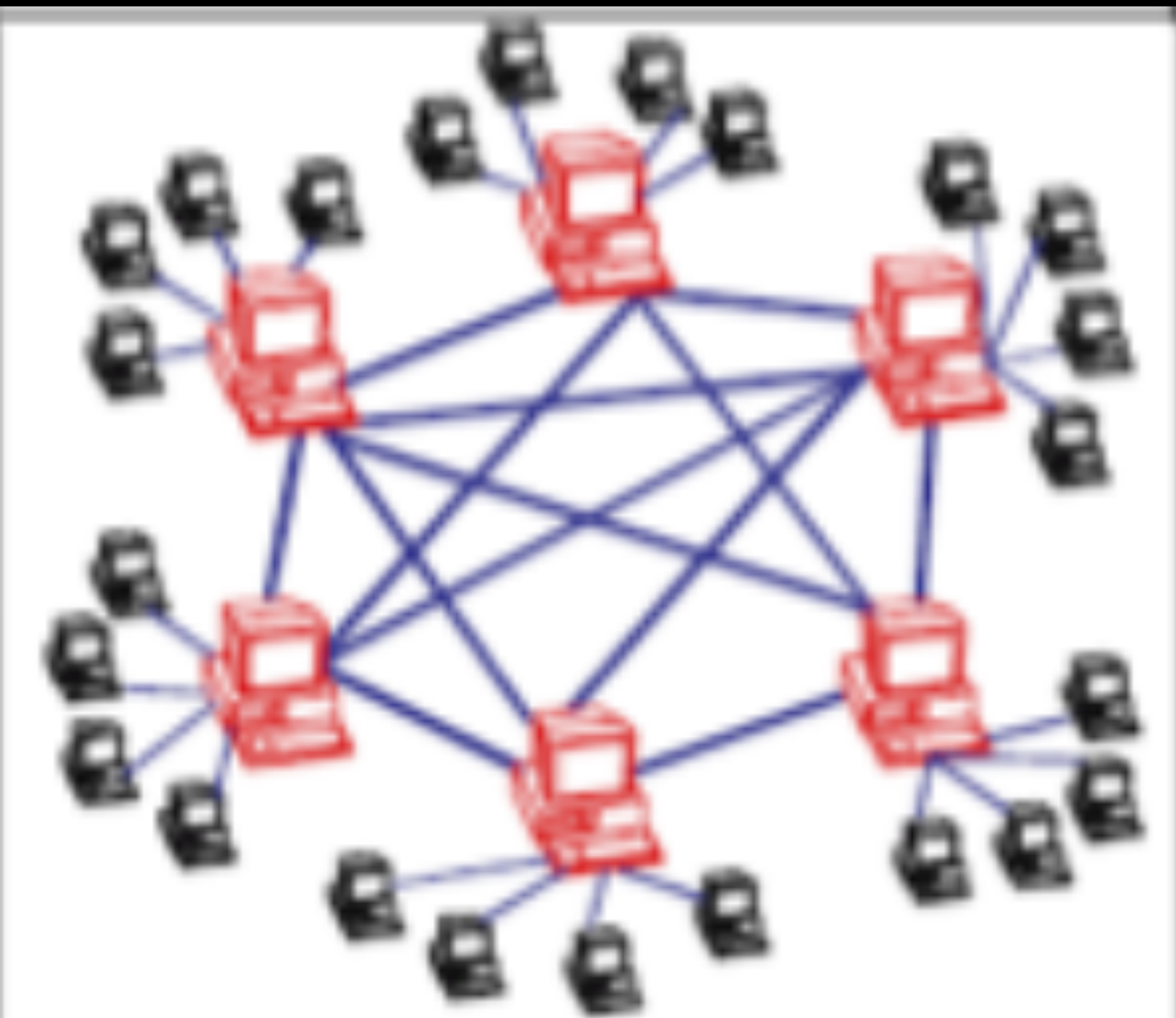




# Pure Peer-to-Peer

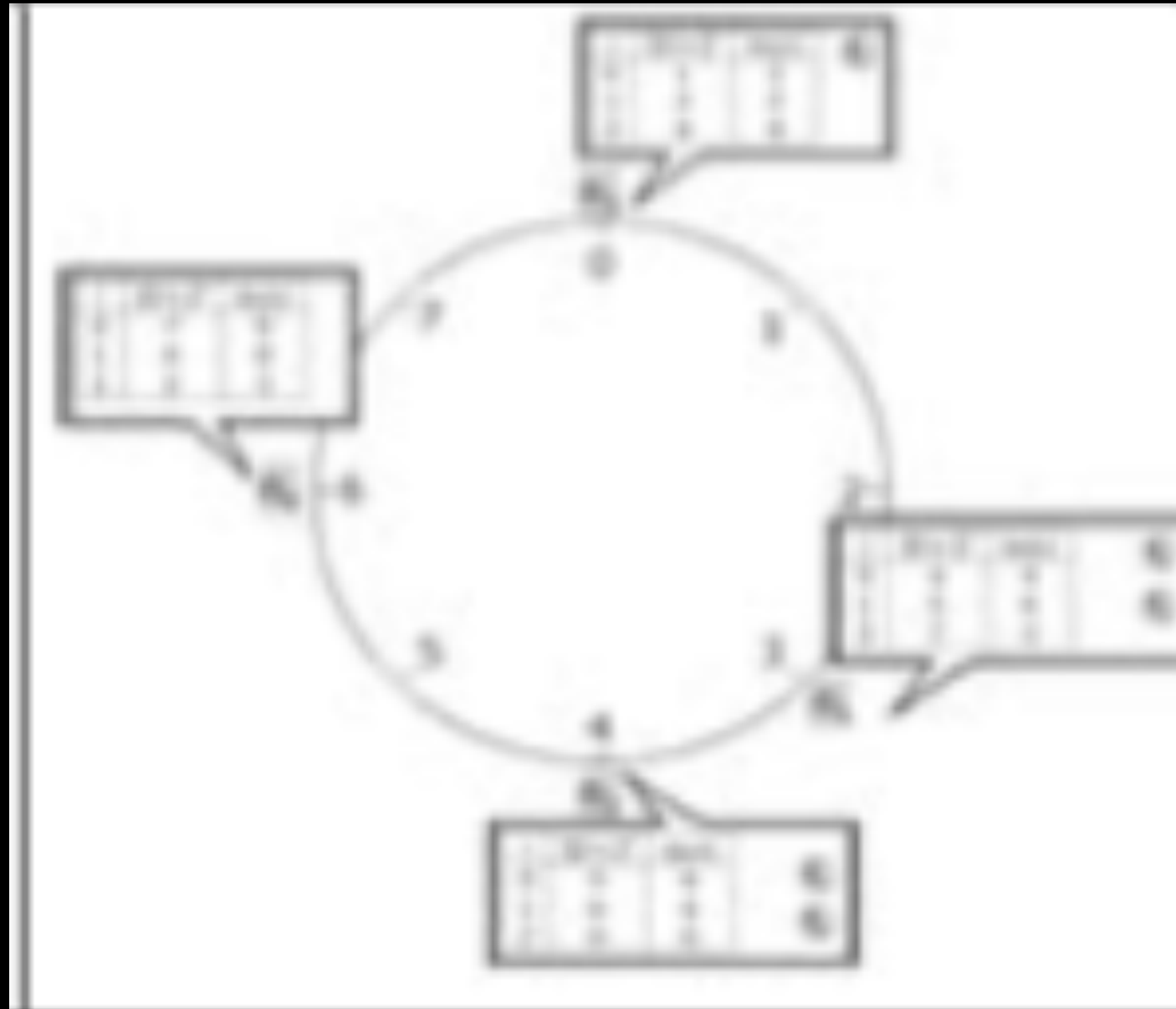


# Hybrid Peer to Peer





# Structured Peer to Peer

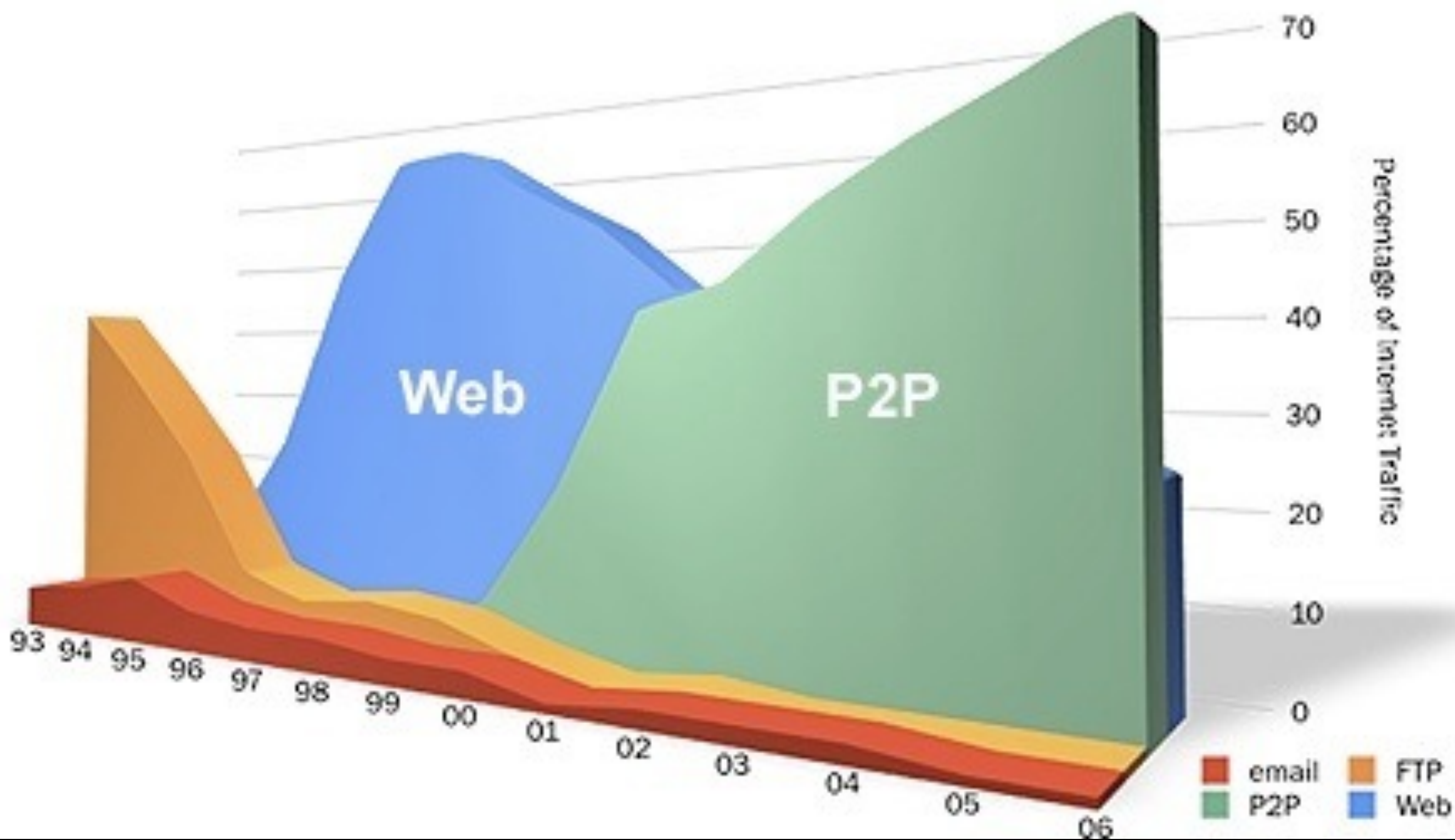


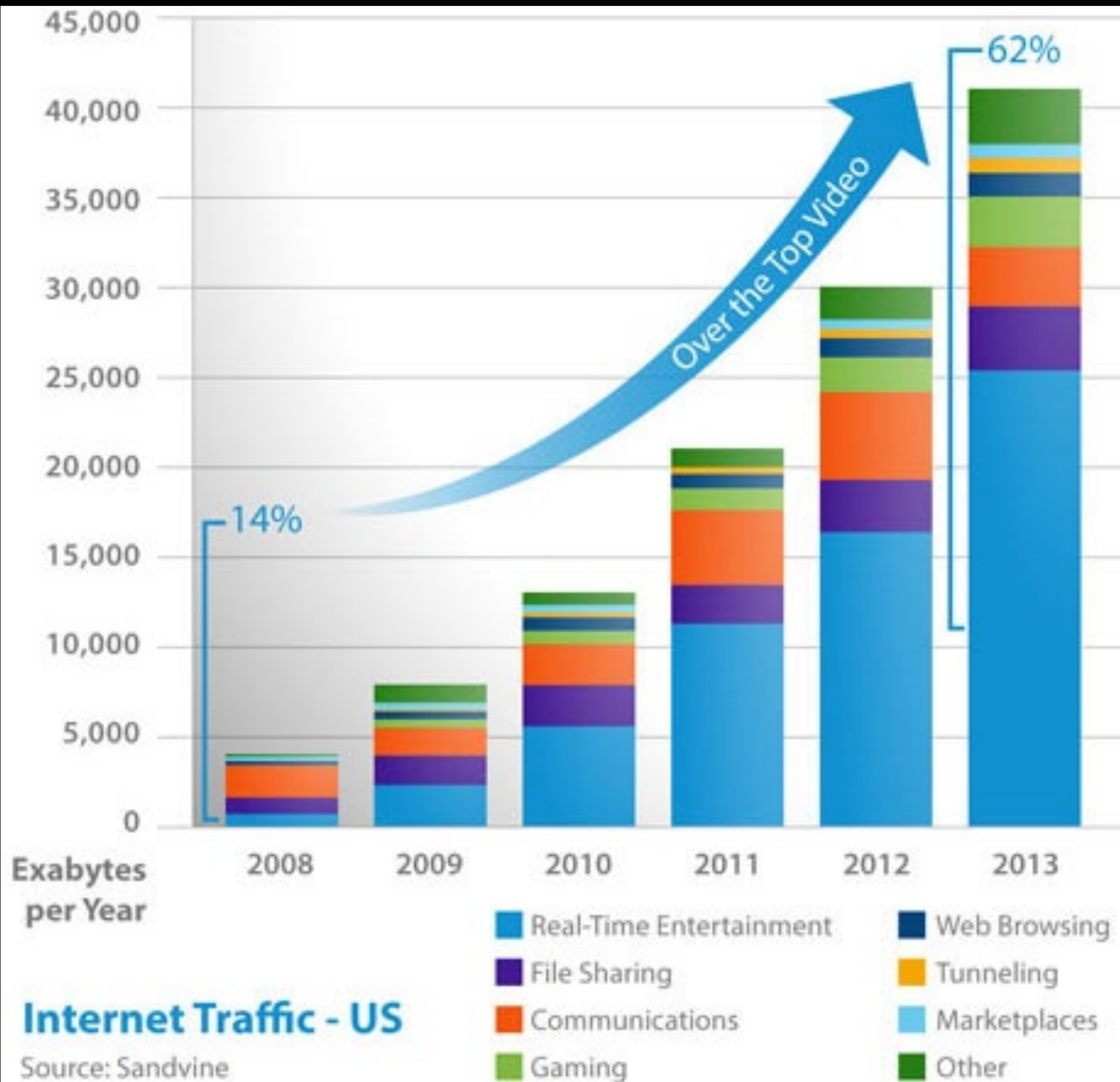
# Client Server

- Centralised
- Easy to develop, analyse and control
- Difficult to scale
- Central Point of Failure

# Peer to Peer

- Decentralised
- Difficult to develop, analyse and control
- Self scaling
- No central point of failure







# Social Issues

# Privacy



# Censorship



The Pirate Bay



# Copyright



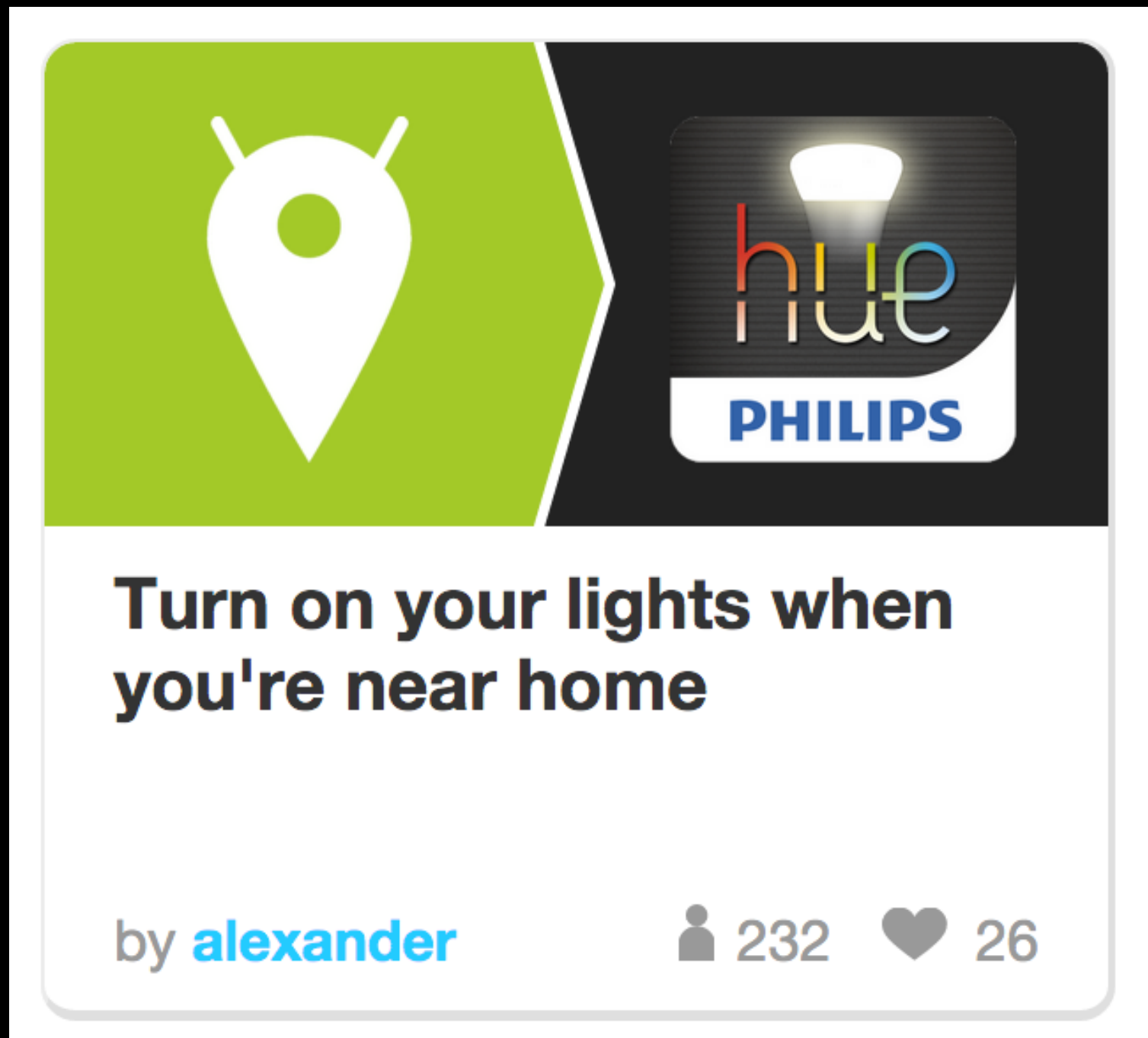
Technology



# BitCoin



# IF This Then That



# Mobile Computing: The Next Decade (2010)



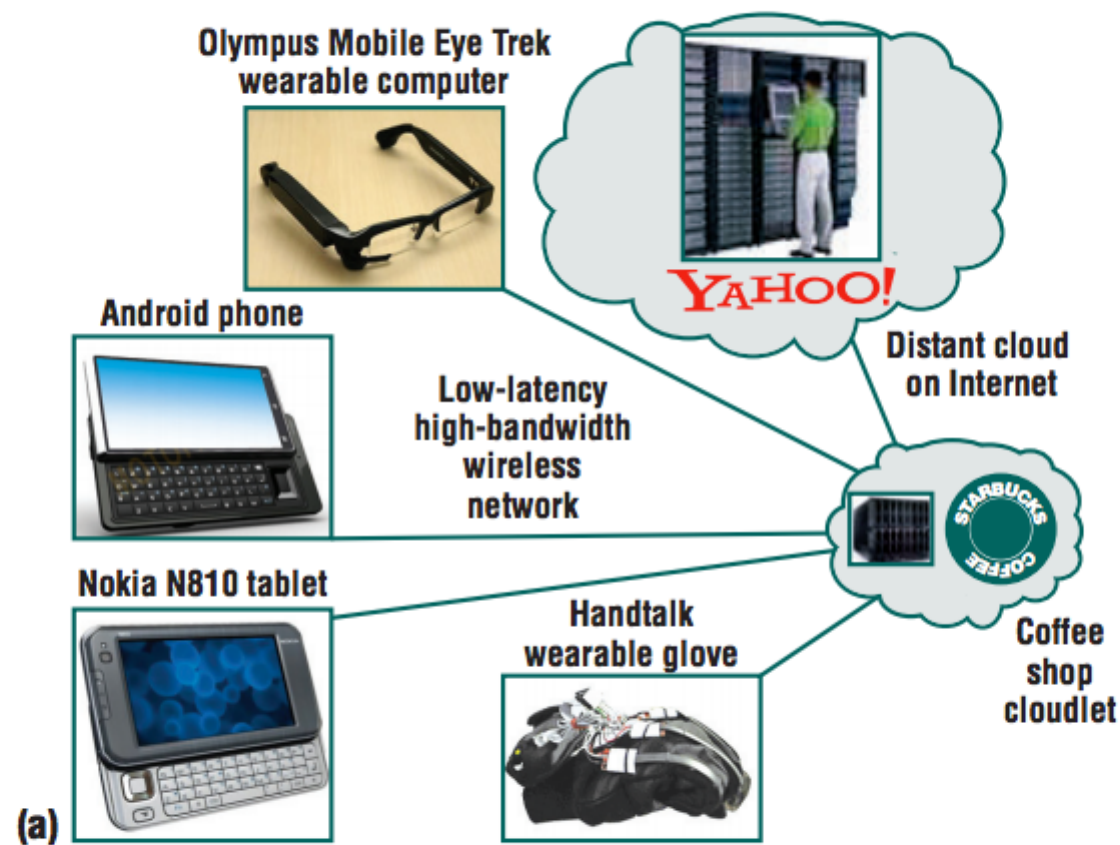
Mahadev Satyanarayanan

# Airplay





# Cloudlets



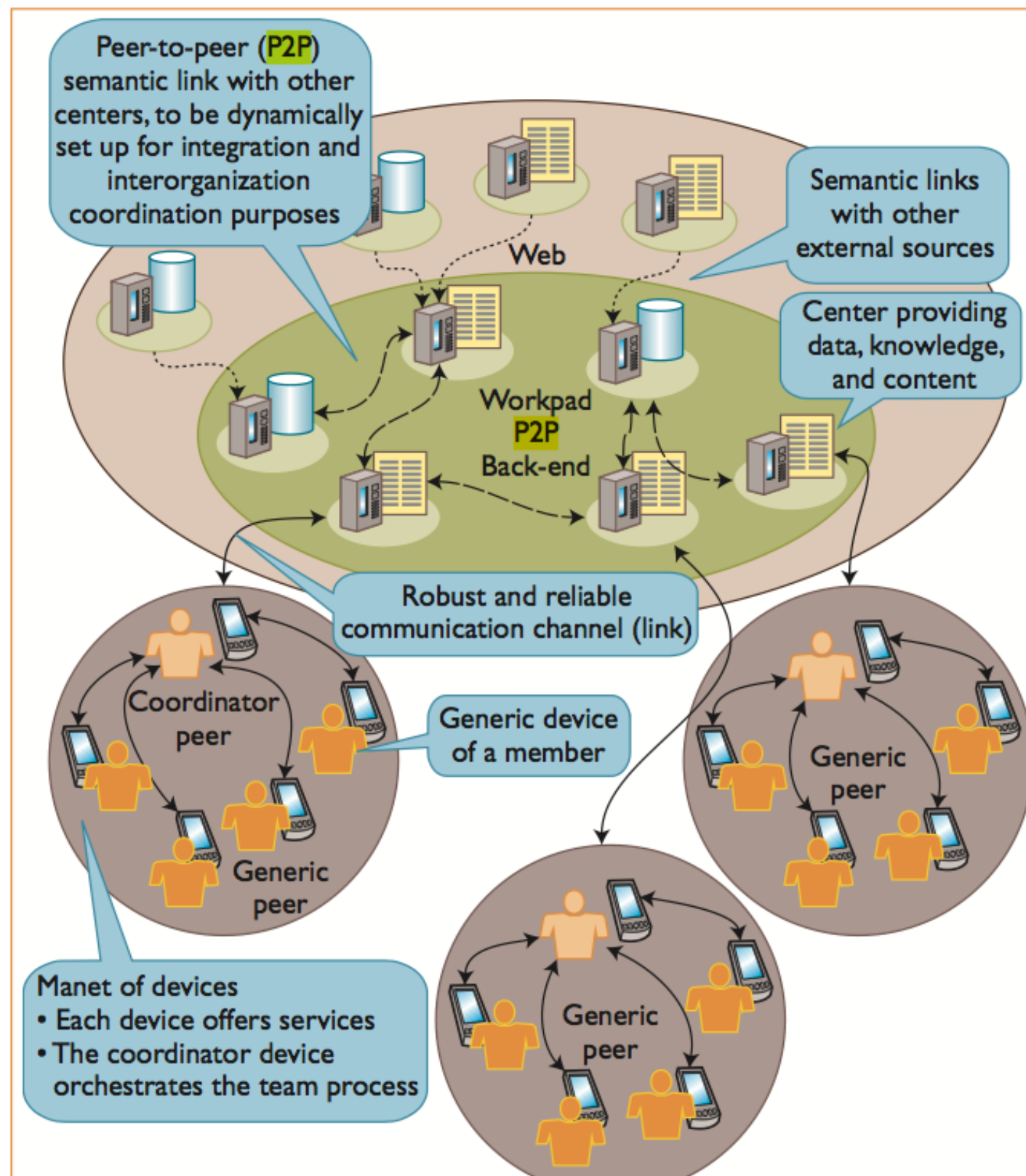
(b)

	Cloudlet	Cloud
State	Only soft state	Hard and soft state
Management	Self-managed; little to no professional attention	Professionally administered, 24/7 operator
Environment	"Datacenter in a box" at business premises	Machine room with power conditioning and cooling
Ownership	Decentralized ownership by local business	Centralized ownership by Amazon, Yahoo, etc.
Network	LAN latency/bandwidth	Internet latency/bandwidth
Sharing	Few users at a time	Hundreds to thousands of users at a time

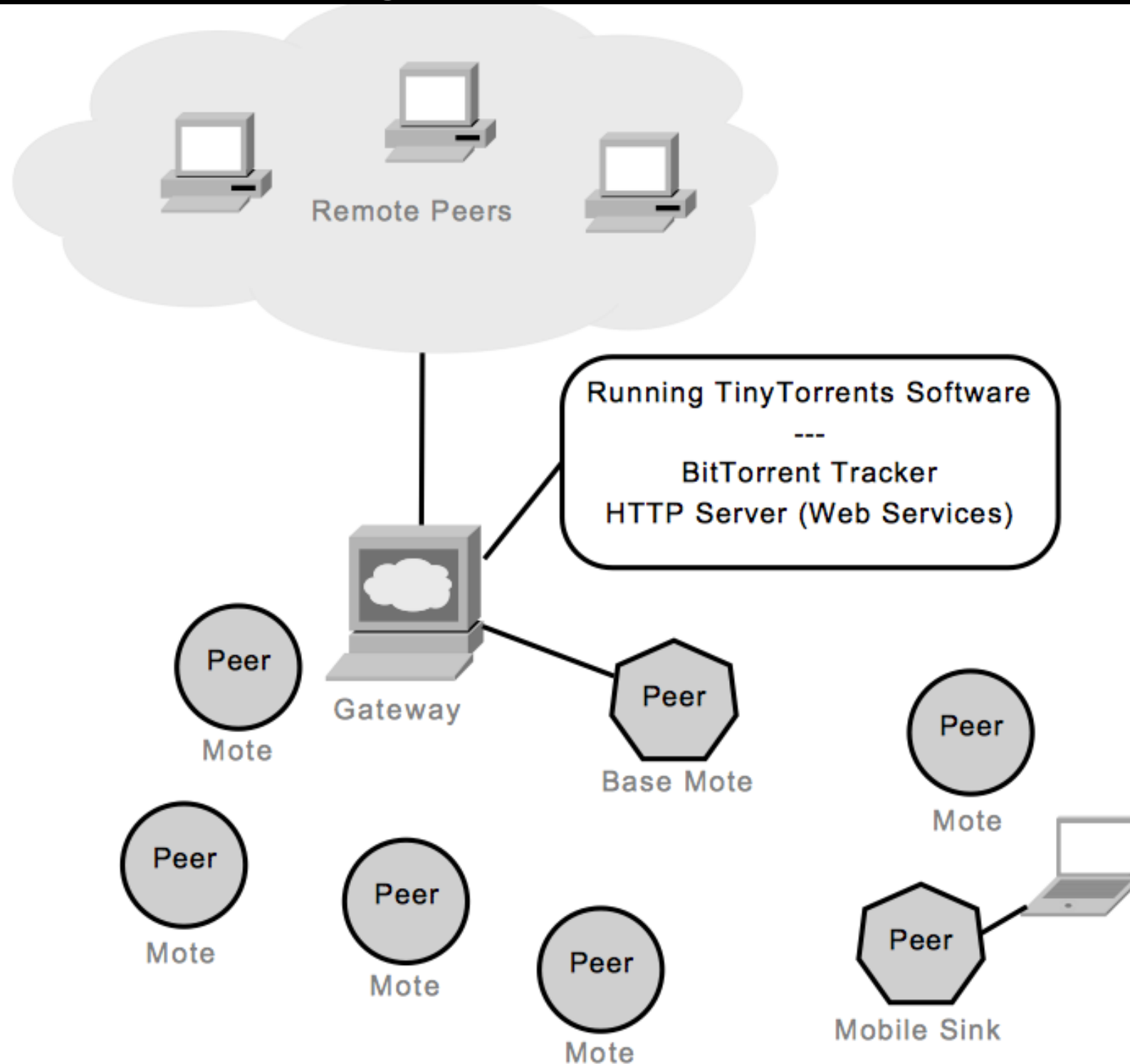
Figure 4. What is a cloudlet? (a) The cloudlet concept involves proximate computing infrastructure that can be leveraged by mobile devices; it has (b) some key differences with the basic cloud computing concept.



# Workpad



# Tiny Torrents



In Conclusion

Thank you