Hybrid Federated Learning on the Cloudy Edges

Oct 6, 2023

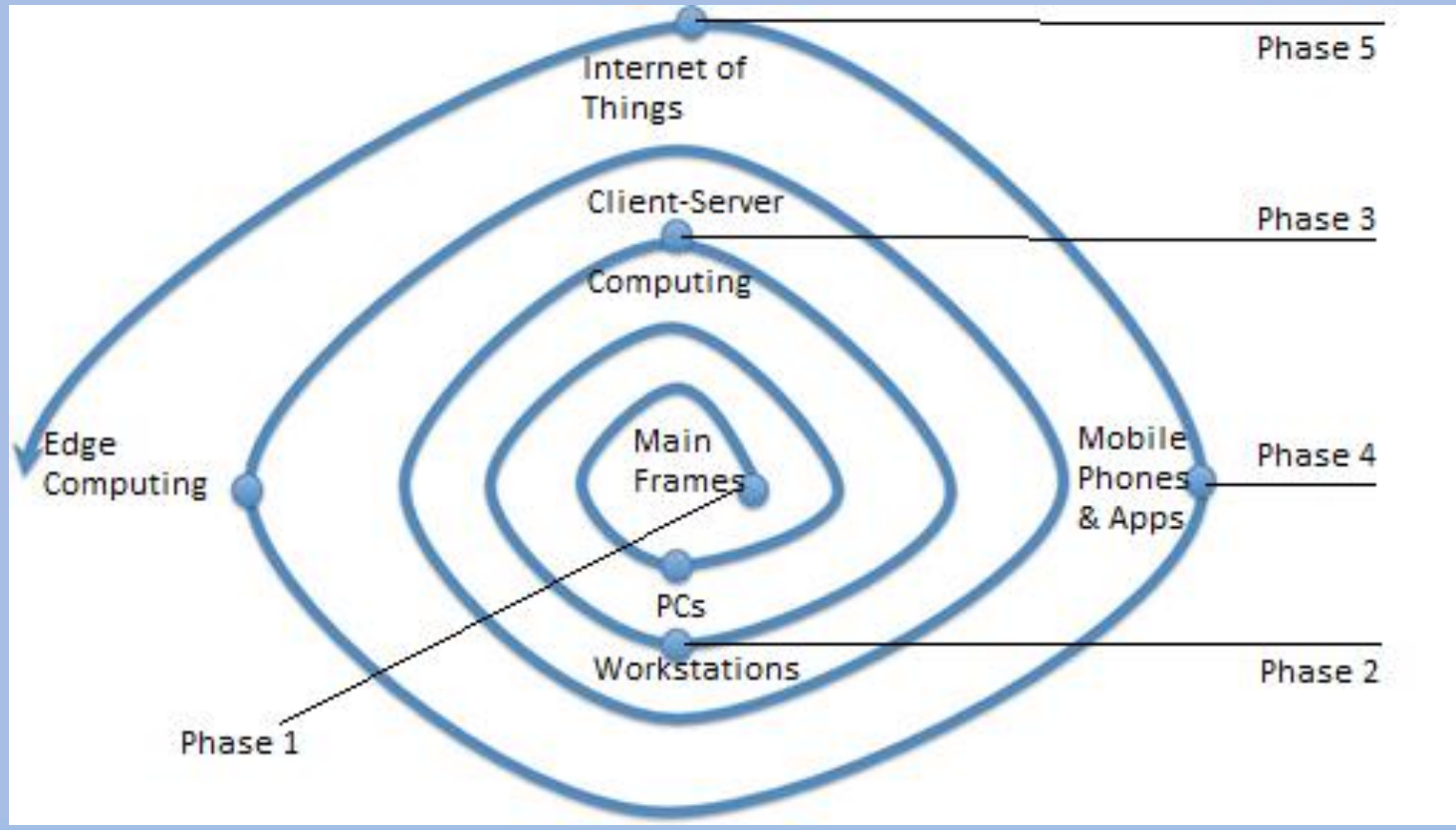
Naresh K. Sehgal, Ph.D.
Partha Pratim Saha

Content

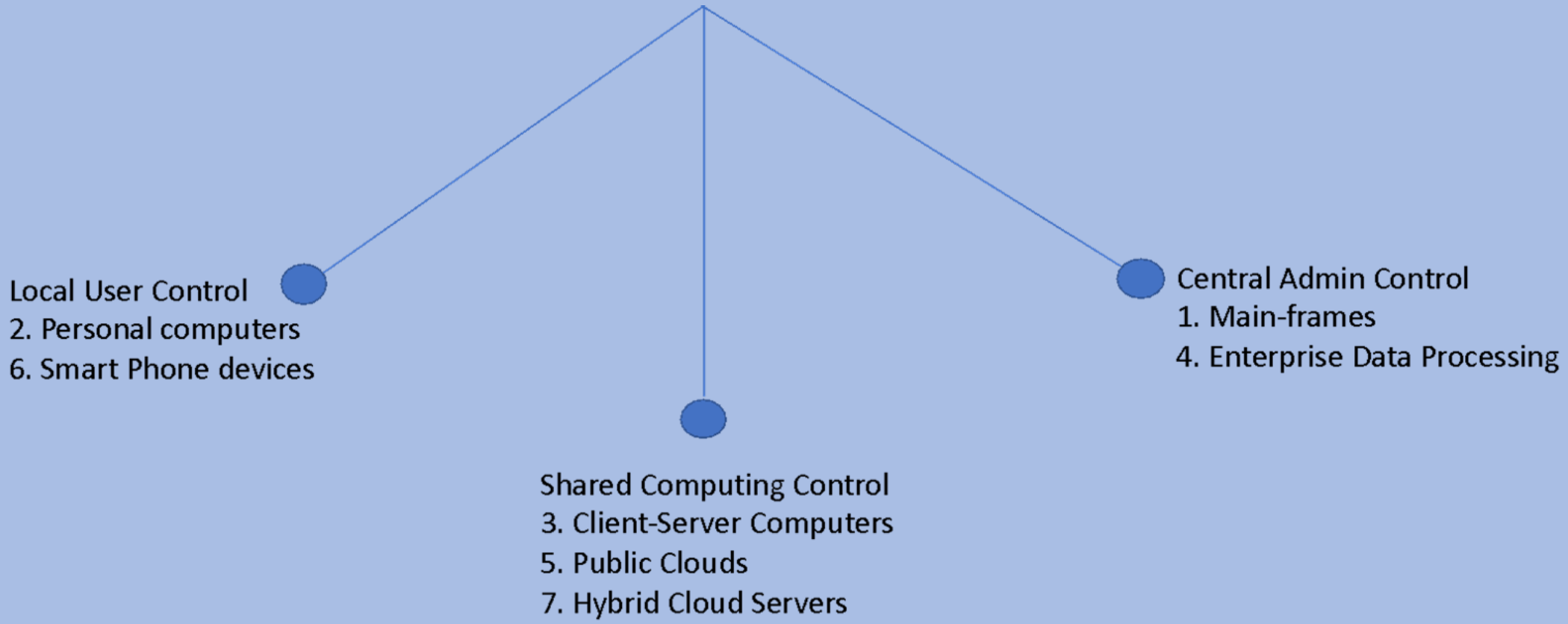
- Why?
- What?
- How?

Why: Evolution of Computing

- Phase 1: Main Frames
- Phase 2: PCs and Workstations
- Phase 3: Client-Server Computing
- Phase 4: Mobile Phones and Apps, supported by hyper-scale DCs
- Phase 5: IOT

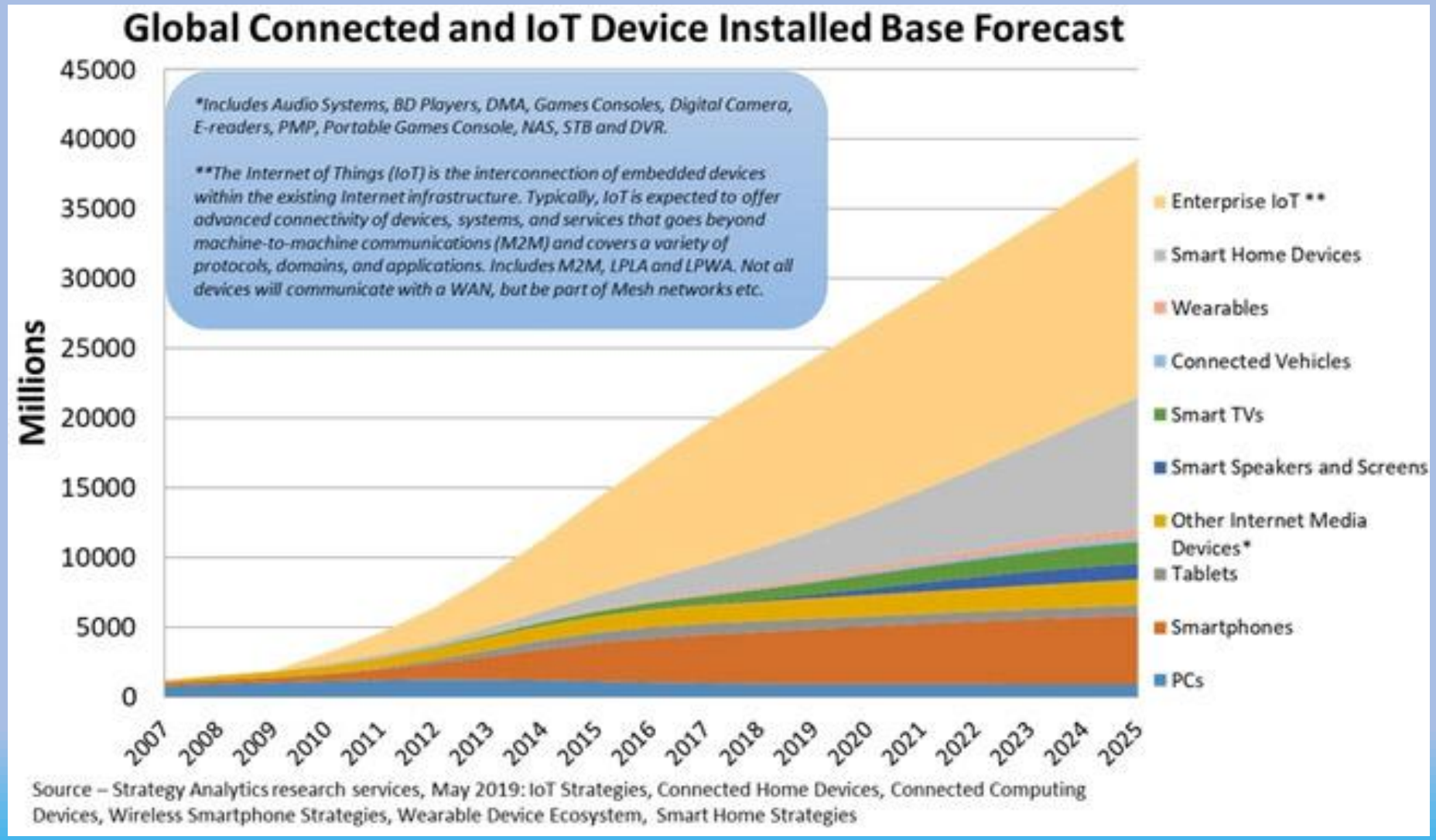


Why (contd): Pendulum of Control is Shifting



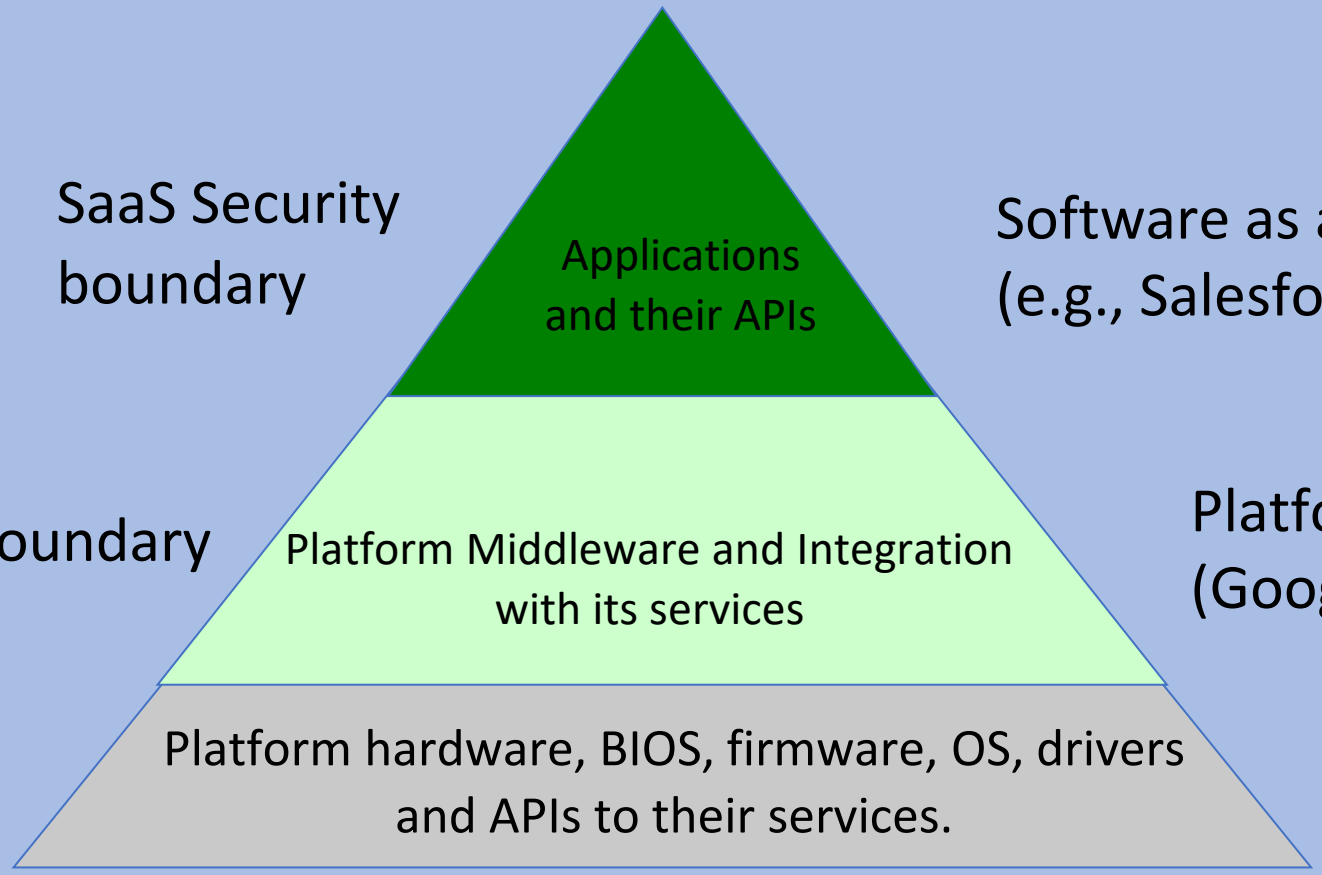
A growing segment of customers want local storage and AI/ML processing, such as hospitals, lawyers and accountants etc.

What: IOT with an Intelligent Compute Node



https://www.helpnetsecurity.com/2019/05/23/connected-devices-growth/

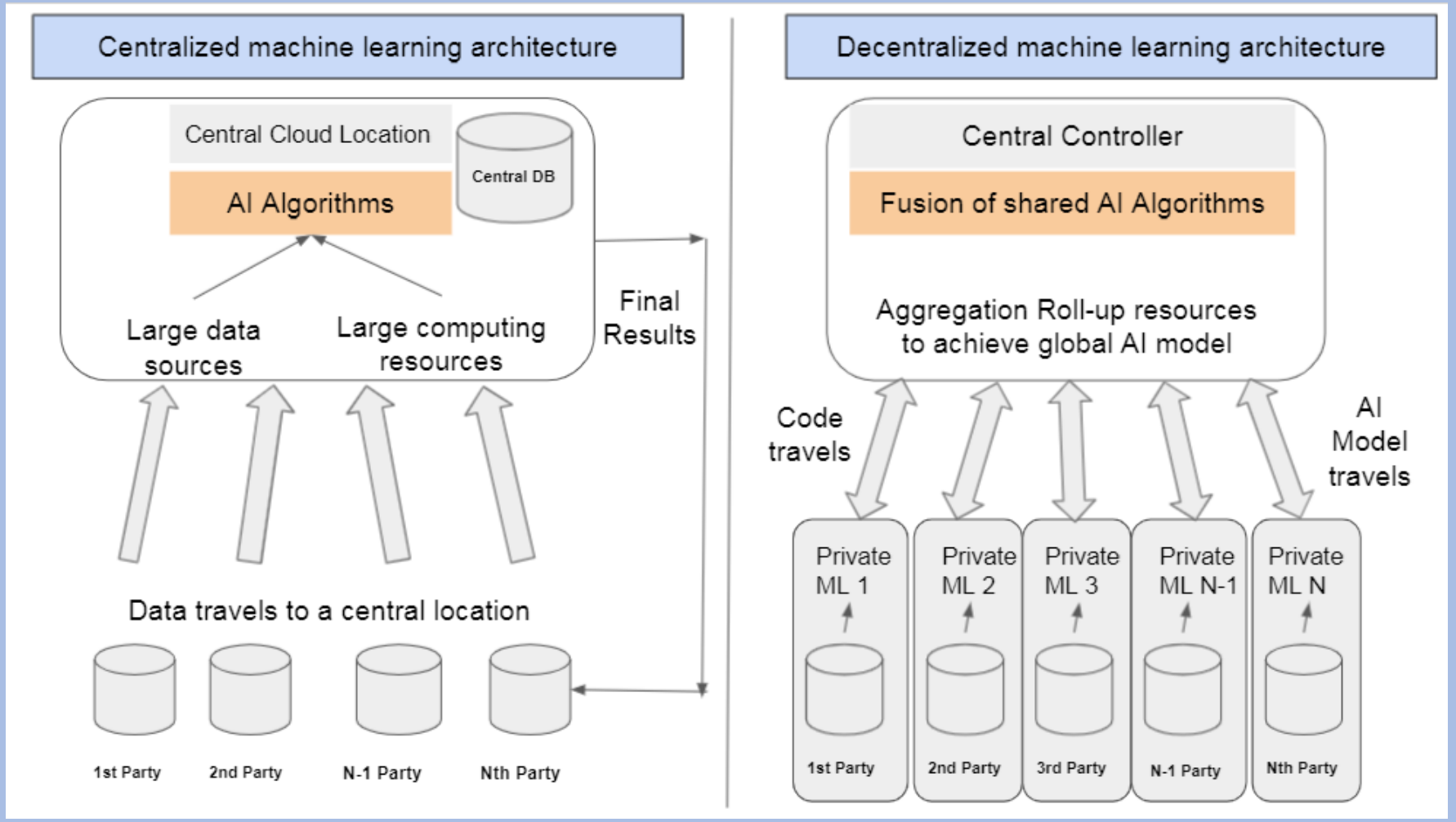
What (contd): Security in Public Clouds



Some Customers with sensitive data are reluctant to use Public Clouds and use On-premise Servers

Centralized and Decentralized Learning

Data is pooled



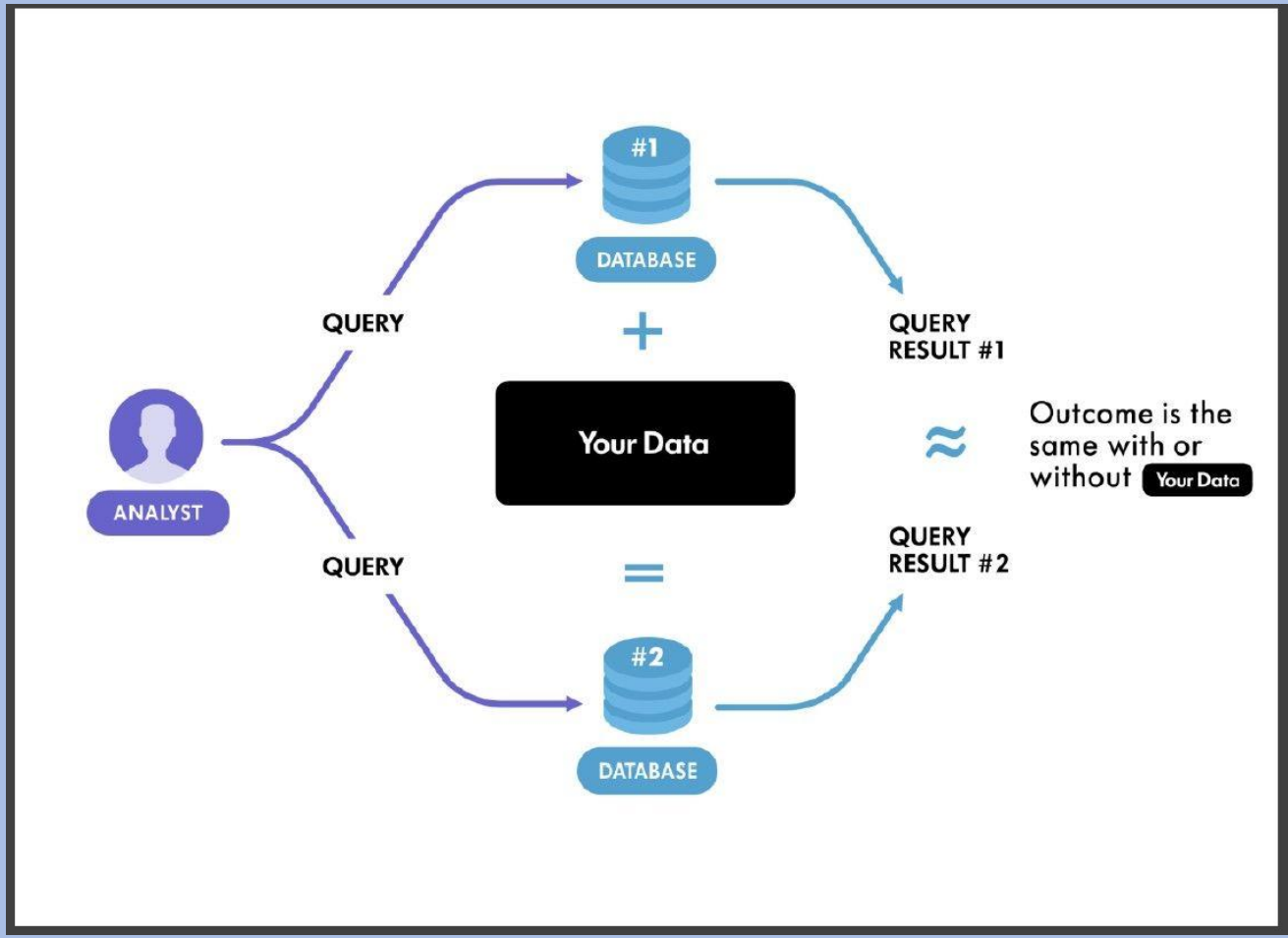
Data stays local

Federated Learning is simply the decentralized form of Machine Learning*

*https://www.analyticsvidhya.com/blog/2021/05/federated-learning-a-beginners-guide/

A Middle Ground: 80-20 rule for Security

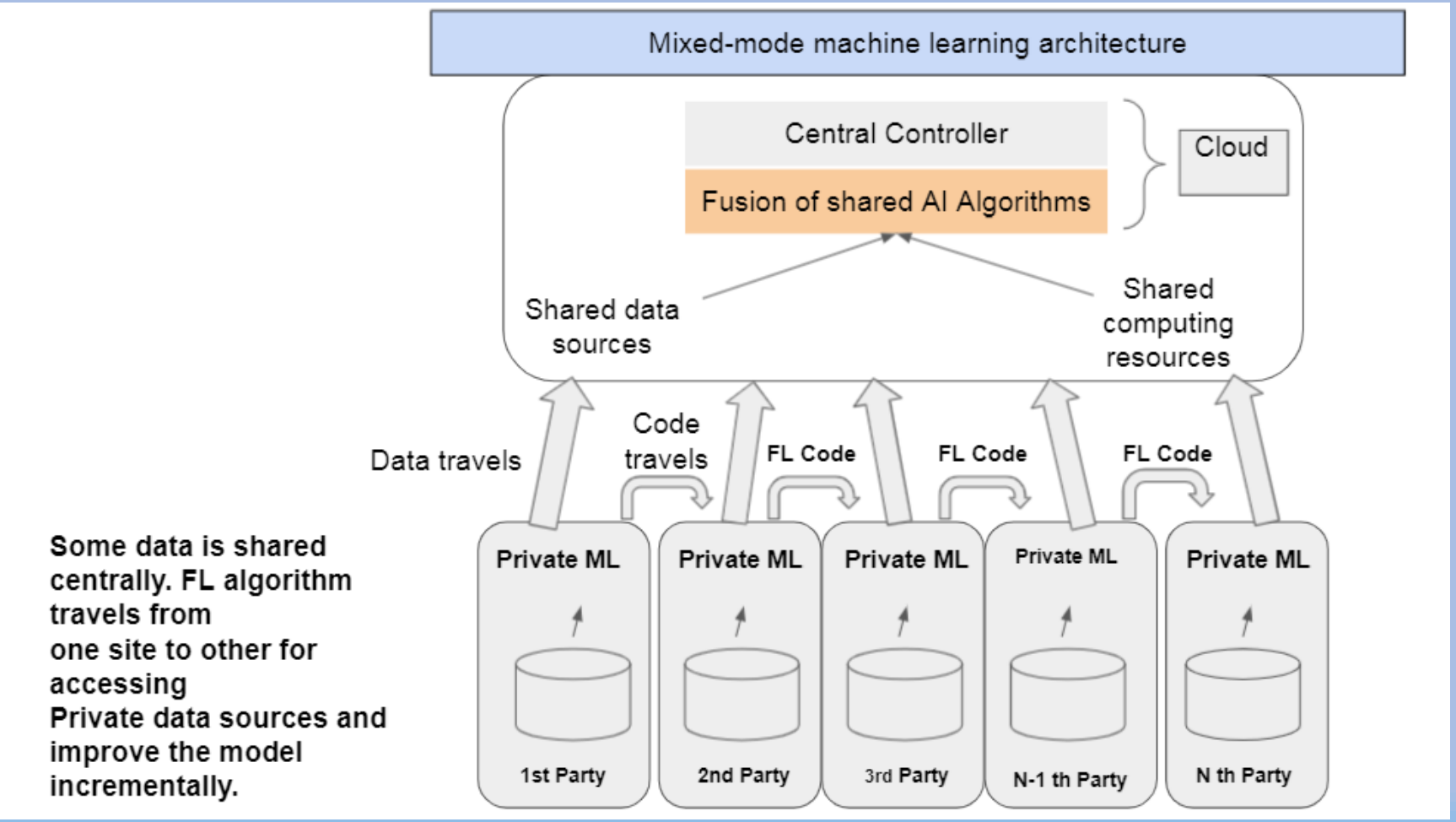
- Some data elements are more critical than others
  - E.g., patient's name, SS#, DOB
- If Private Health Info (PHI) or PII (Personal Identifiable Info) is removed, then rest of data (>80%) can be shared
- PHI or PII can be added back later on, end result is same



Learning nothing about an individual while learning useful information about a population*

*https://blog.openmined.org/differential-privacy-using-pydp/

Hybrid Federated Learning (HFL) Architecture



Differential Privacy: Divide the data in two parts: private and public

HFL Medical Drug Research Study

Consider three entities, a Hospital = A, Drug Company = B, and Medical Researcher = C, with a single centralized server.

Some notations are below:

t_{da} = data copying delays from Hospital A to central server
t_{db} = data copying delays from Drug Company B to central server
t_{dc} = data copying delays from Medical Company C to central server
t_{pa} = time for code and weights of Neural Network to travel from central server to hospital A
t_{pb} = time for code and weights of Neural Network to travel from central server to Drug Company B
t_{pc} = time for code and weights of Neural Network to travel from central server to Medical Researcher C
t_{px} = program execution time

n = number of training iterations

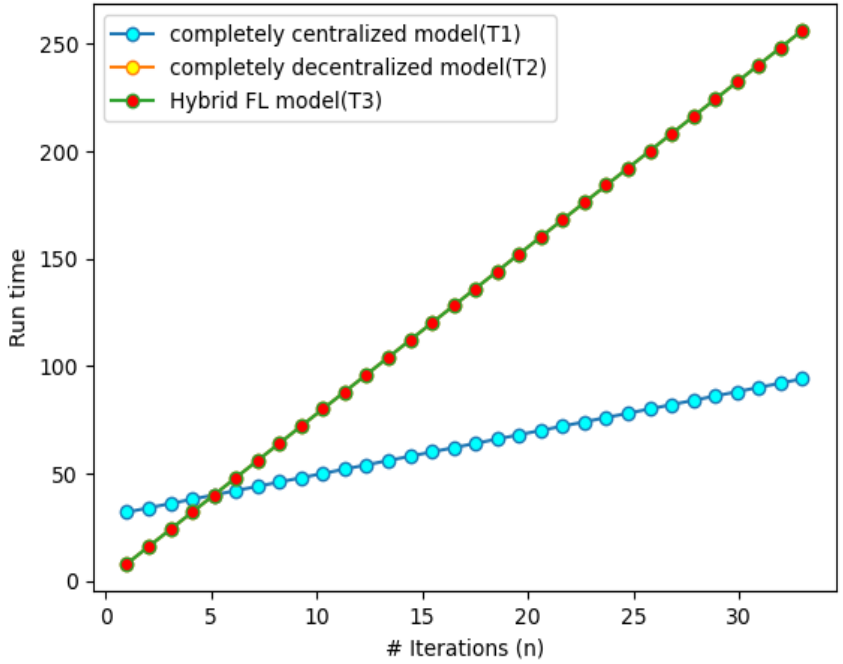
So, total worst case (asynchronized) data copy time to central database is = $t_{da} + t_{db} + t_{dc}$

and in a completely centralized model, total worst case run time will be $T1 = t_{da} + t_{db} + t_{dc} + n * t_{Px}$

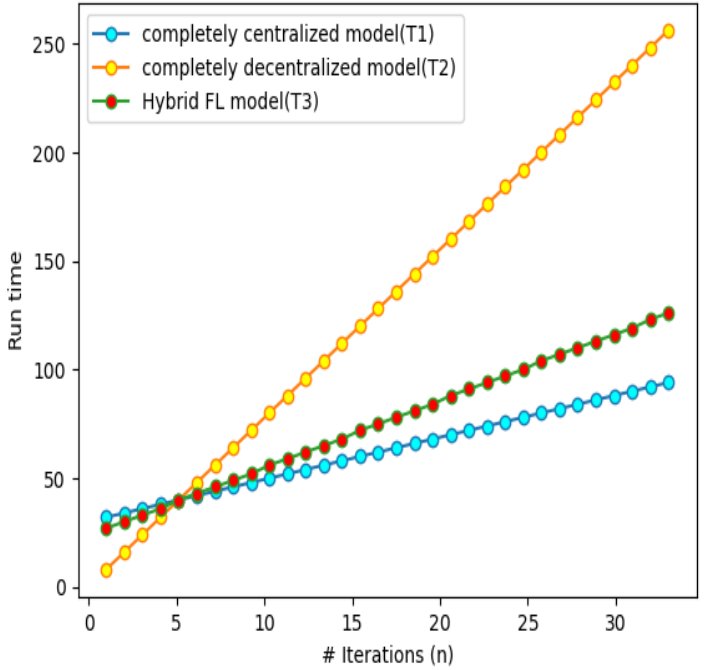
For a fully decentralized Federated learning system, total worst run time will be $T2 = n * (t_{pa} + t_{pb} + t_{pc} + t_{px})$

For larger n, $T2 >> T1$, because in T1 we copy data once, whereas in T2, program has to travel every iteration

Program size is smaller than dataset size with 0% data share : Hybrid = de-centralized

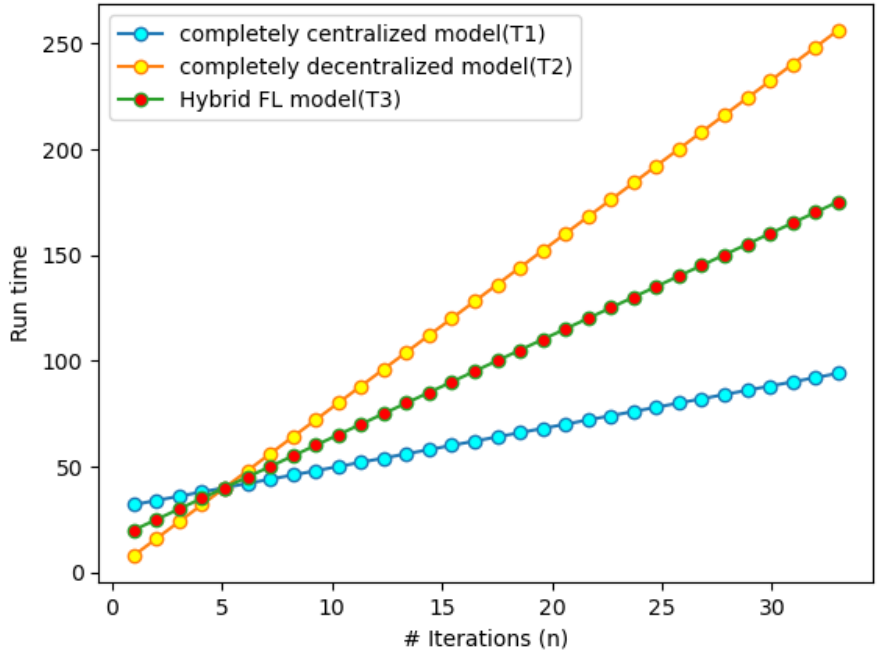


Program size is smaller than dataset size with 80% data share : Hybrid is even more towards centralized

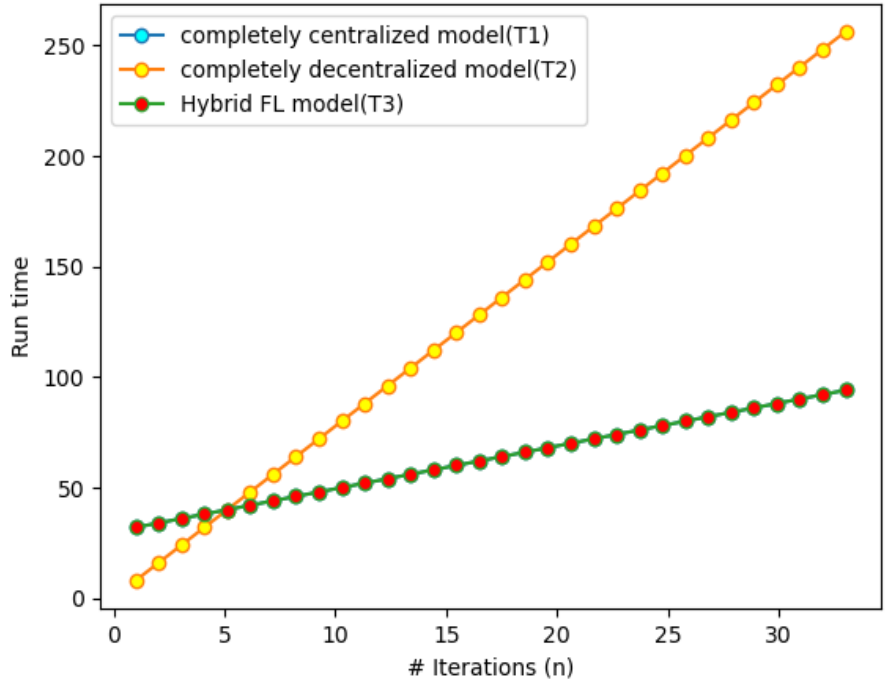



```
tda = 5
tdb = 10
tdc = 15

tPa = 1
tPb = 2
tPc = 3

tPx = 2
```

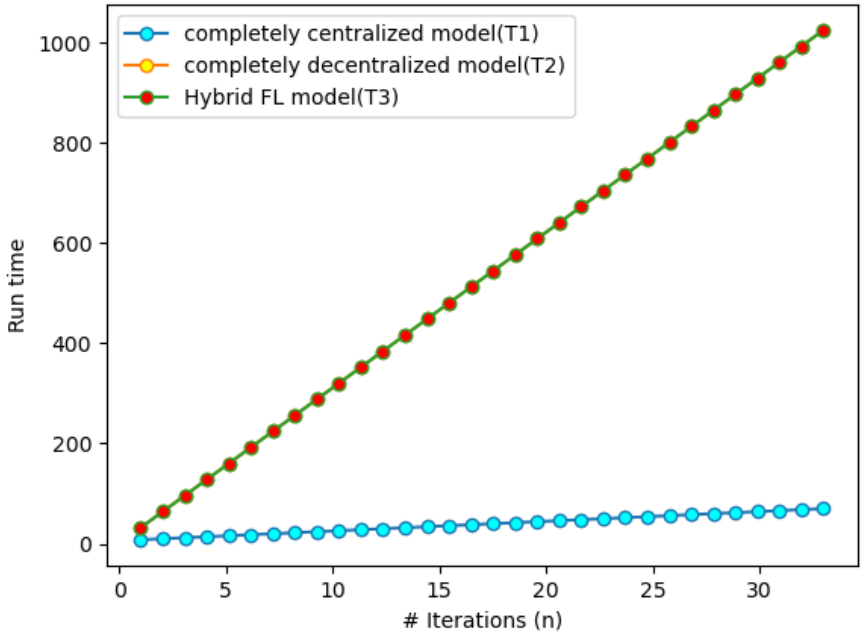
Program size is smaller than dataset size with 50% data share : Hybrid is in the middle



Program size is smaller than dataset size with 100% data share : Hybrid = centralized

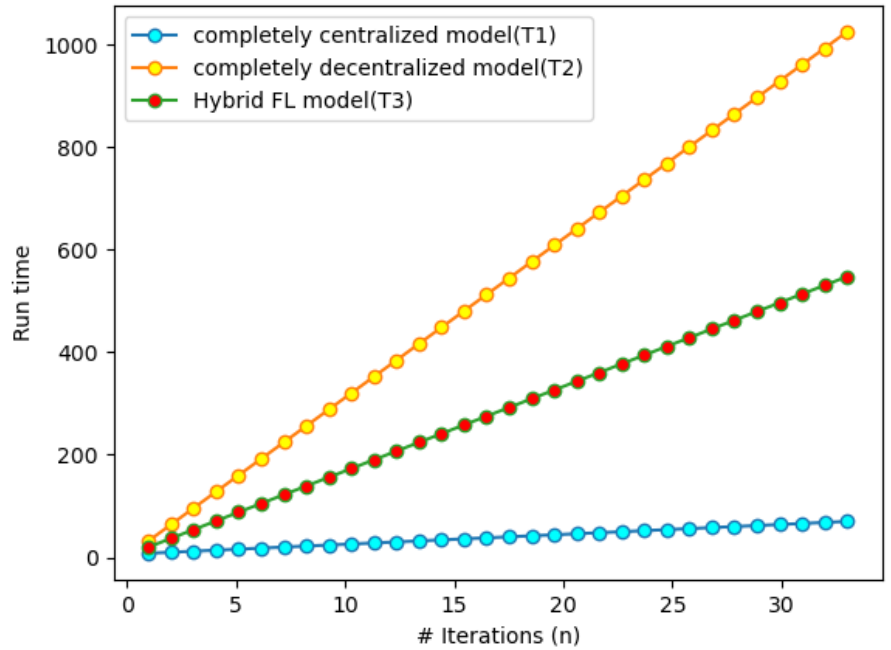


Program size is bigger than dataset size with 0% data share : Hybrid = De-centralized

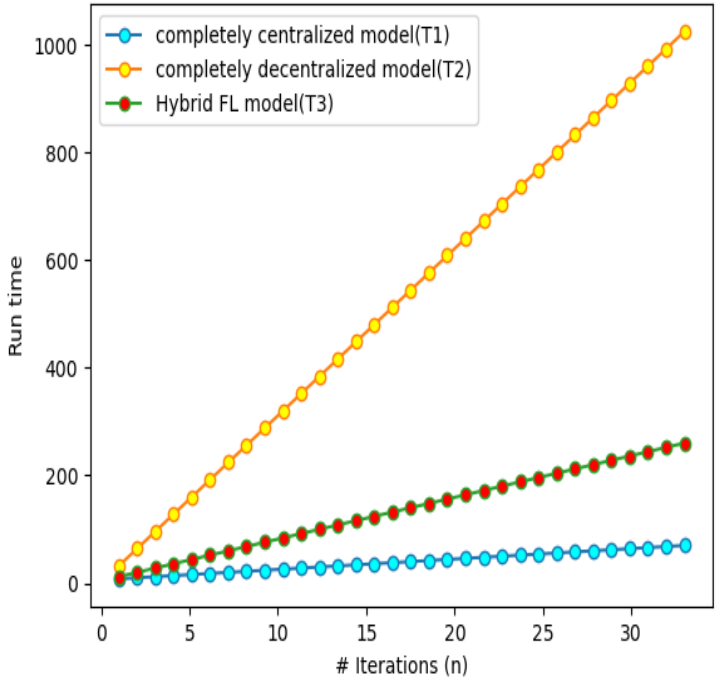



```
tda = 1
tdb = 2
tdc = 3

tPa = 5
tPb = 10
tPc = 15

tPx = 2
```

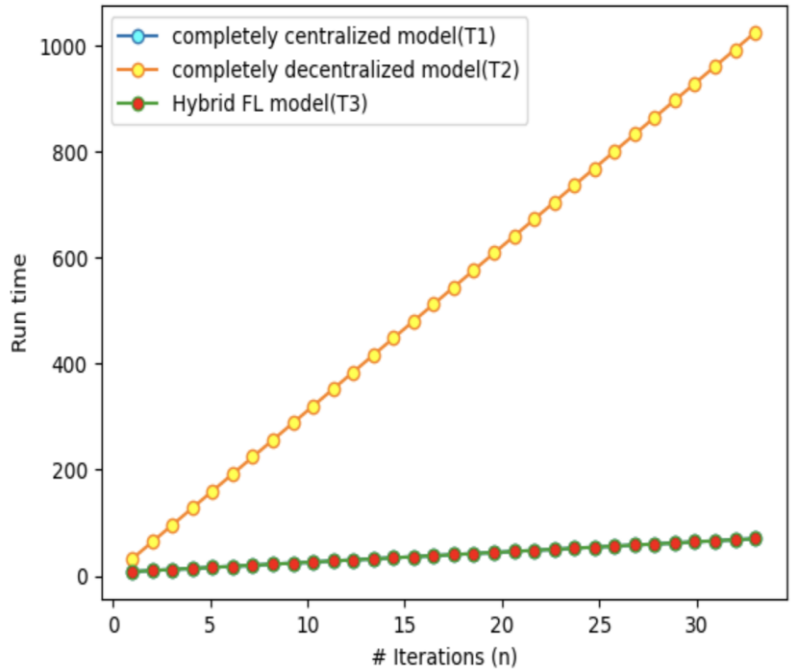
Program size is bigger than dataset size with 50% data share : Hybrid is in the middle



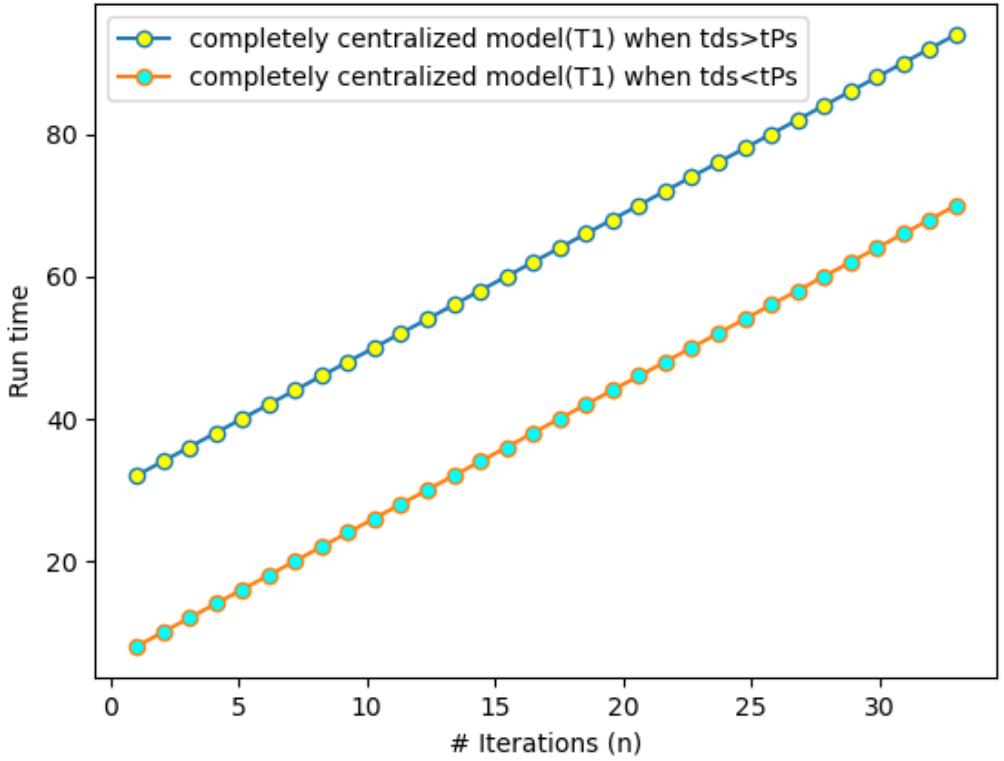
Program size is bigger than dataset size with 80% data share : Hybrid is even more towards centralized



Program size is bigger than dataset size with 100% data share : Hybrid = centralized

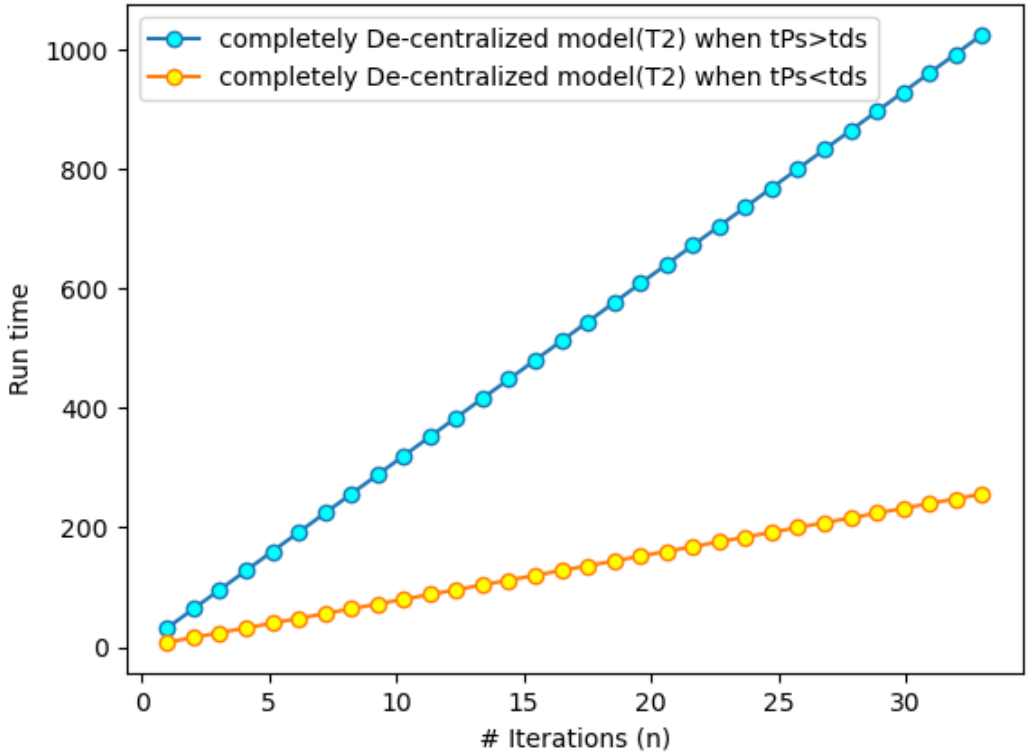


Centralized model when dataset size is greater than program size and vice-versa



Which is better and why?

De-centralized model when dataset size is greater than program size and vice-versa

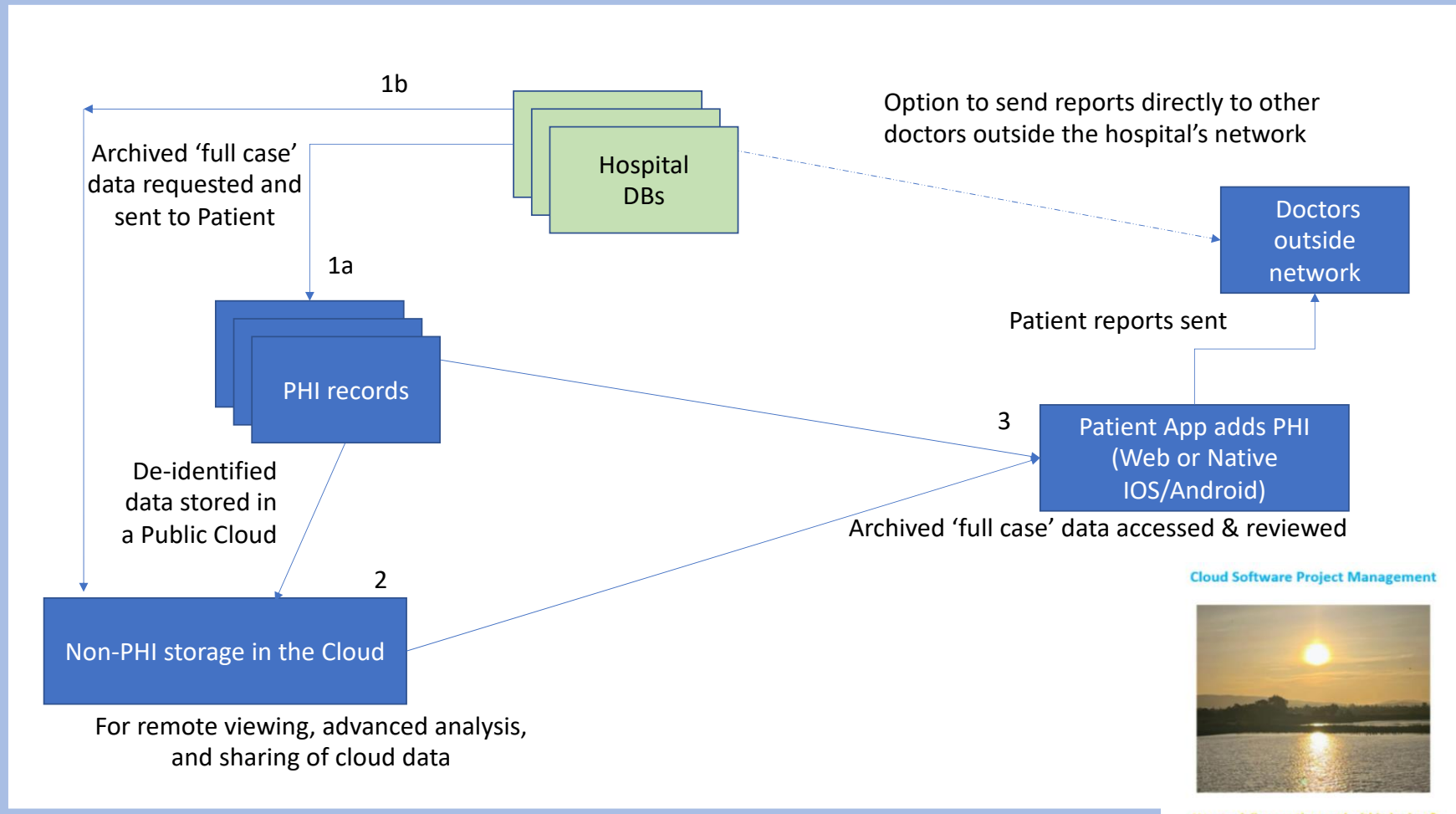


Centralized is always faster

Use HFL with Privacy Preserving Analytics at the Edge of a Network

Idea for a new Patient facing Application*

1) Patients can access their own medical data when desired
2) Request each hospital they visit and interact with to release medical records.
3) Patients can control who else can access their data for reading or updating
4) App provider can share non-PHI data in a public Cloud with other entities for potential monetization
5) Analytics on non-PHI data can help Patients, Doctors and Medical Researchers



*https://a.co/d/1k3d1xl

Edge Computing Security Challenges

- Definition of a Cloud has been expanding, getting out of a data center
- Perimeter defense is insufficient, as there is no fixed perimeter
- Fixed protocols for boundaries of security fail, shared security model
- A fixed universal security policy is inadequate, each party owns their data
- Resources on Edge need to be adaptive, for varying amount of compute

Conclusions

1. **If de-centralized configuration is slower**
   - **Then share more data, but keep private data on-site for security**

2. **If centralized configuration is slower**
   - **Then keep all/more data on-site, and use Edge Computing**
   - **Better for both security and performance**

3. **Hybrid FL Challenges**
   a) **Data sharing considerations:** Honest sharing and Security concerns
   b) **Managing incremental data changes:** Keeping all parties in synch
   c) **Local vs. global ML models**: Performance vs. Accuracy tradeoffs