‘Networks of Things’

Pieces, Parts, and Data

J. Voas
Computer Scientist
US National Institute of Standards and Technology

jeff.voas@nist.gov
j.voas@ieee.org

Problem

What is IoT?

Opening Statement

A Network of Things (NoT) or ‘subnet of things’ employs a mixture of sensing, communication, computation.

A Network of Things (NoT) or ‘subnet of things’ leads to actionable decisions or predictions. Things may be *private* or *public*. Things may be 3rd party or homegrown.

Wikipedia Definition

*The **Internet of Things** (**IoT**) is the interconnection of uniquely identifiable embedded computing devices within the existing Internet infrastructure.*

IEEE's IoT Initiative Definition

- from the Initiative's White Paper

- **Small environment scenario:**
  - It's a network that connects uniquely identifiable *"Things"* to the internet.
  - The "Things" have sensing/actuation and potential programmability capability.
  - Information about the "Thing" can be collected.
  - The state of the "Thing" can be changed.
  - From anywhere, at anytime, by anything
- **Large environment scenario:**
  - A self-configuring and adaptive complex network that interconnects *"things"* to the internet through the use of interoperable communication protocol.

NIST Challenge

- from National Security Telecommunications Advisory Committee (NSTAC) Report on IoT

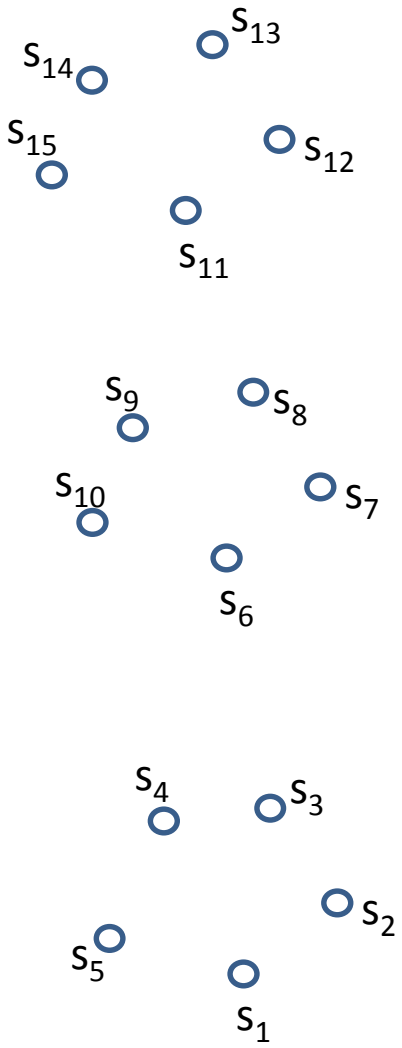
- *Direct the Department of Commerce, specifically NIST, to develop a definition of IoT for use by departments and agencies to be used during assessments related to the IoT.*

- http://www.dhs.gov/sites/default/files/publications/IoT%20Final%20Draft%20Report%2011-2014.pdf

Ten Primitives

1. Sensor
2. Time snapshot (time)
3. Cluster
4. Concentrator
5. Weight
6. Communication channel
7. *e*Utility
8. Decision
9. Geographic location
10. Owner

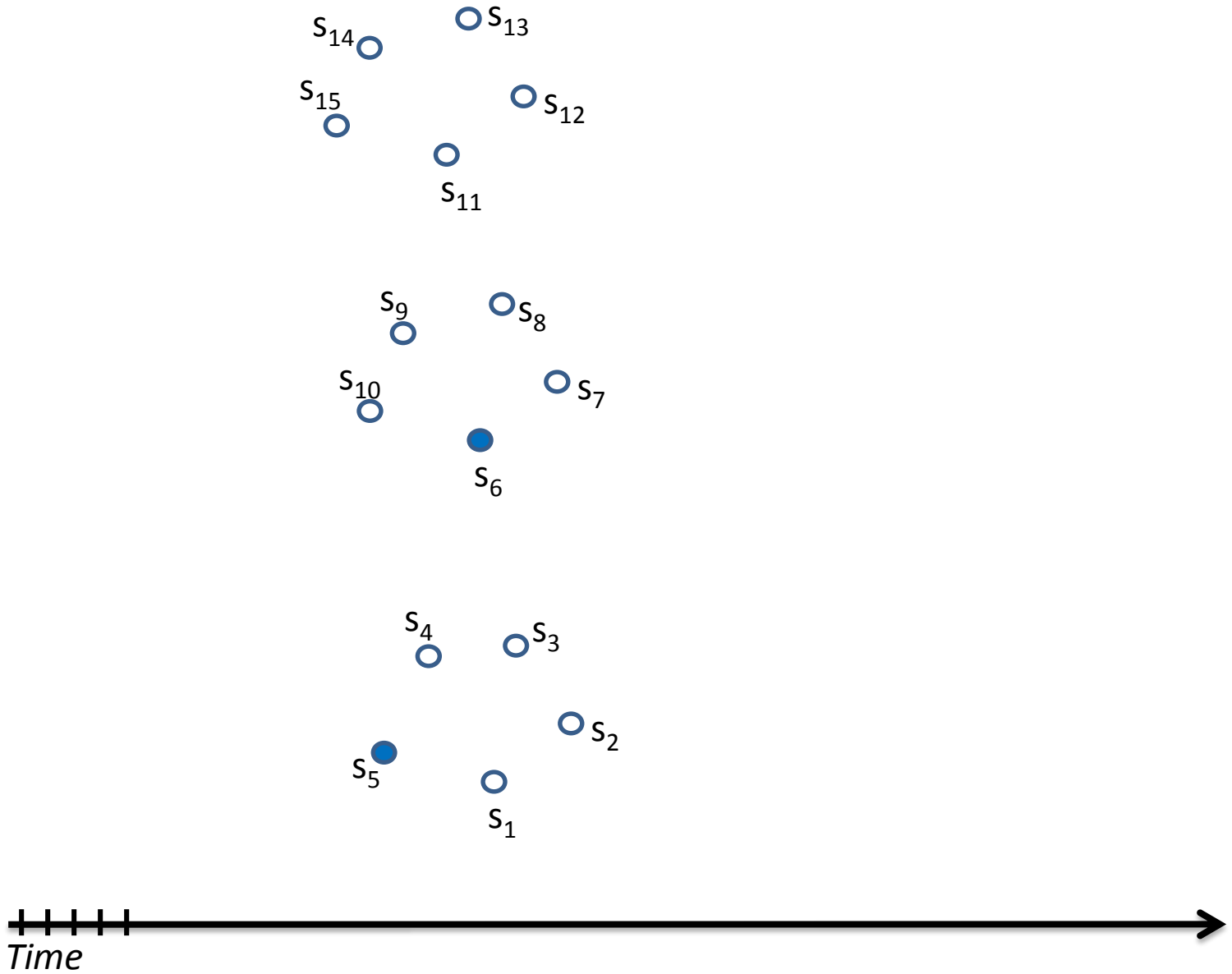
Figure 0: Sensors



Sensor

1. Basic sensors will have little or no software functionality and computing power; more advanced sensors may have software functionality and computing power
2. Sensors will be heterogeneous
3. Sensors have operating geographic locations that can change
4. Sensors may have owner(s)
5. Sensors have pedigree – geographic locations of origin and manufacturers. Pedigree may be unknown or suspicious
6. Sensors may fail or fail intermittently
7. Most sensors are assumed to be cheap, disposable, and susceptible to wear-out over time; building security into a specific sensor will be rarely cost effective
8. Sensors may return no data, totally flawed data, partially flawed data, or correct/acceptable data
9. Sensor repair is usually handled by replacement
10. Each sensor can have a level of data integrity ascribed to it
11. Sensors will have their data tokenized to void security concerns. We assume tokenization (encryption) is correct and immune to compromise
12. Sensors and/or their data may be leased to multiple NoTs concurrently. A sensor can have one or more recipients' of its data
13. The frequency with which sensors release data impacts the data's integrity.
14. Data can become *stale*
15. Reliability is a concern for 3rd party sensors
16. Possible sensor tampering is a security concern.

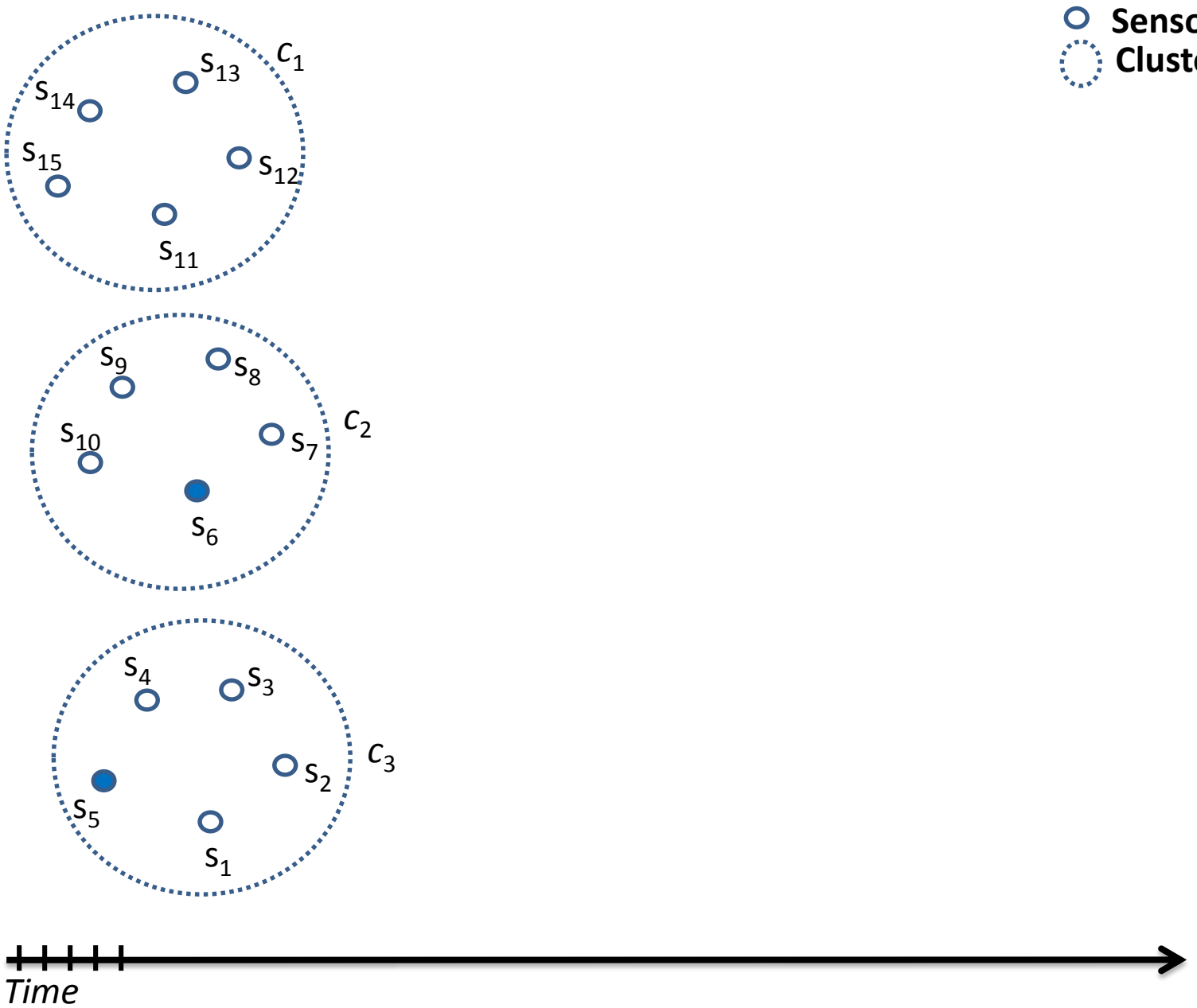
Figure 1: Snapshots



Snapshot

1. Sensors release data that is either event-driven or released at pre-defined time snapshot intervals
2. NoTs may affect net neutrality – sensing, communicating, and computing can speed-up or slow-down a NoT's workflow

Figure 2: Clusters



Cluster

1. Clusters are groupings of sensors that can appear and disappear instantaneously resulting in potential late-binding
2. Clusters are abstractions
3. Clusters are not physical
4. C_i is a *cluster* of n ≥ 2 sensors, $\{s_1, s_2, s_3, ..., s_n\}$
5. C_i may share a sensor with C_k, where i ≠ k
6. *Late-binding* of a sensor to a cluster may result in little ability to mitigate trustworthiness concerns

Figure 3a: Concentrators

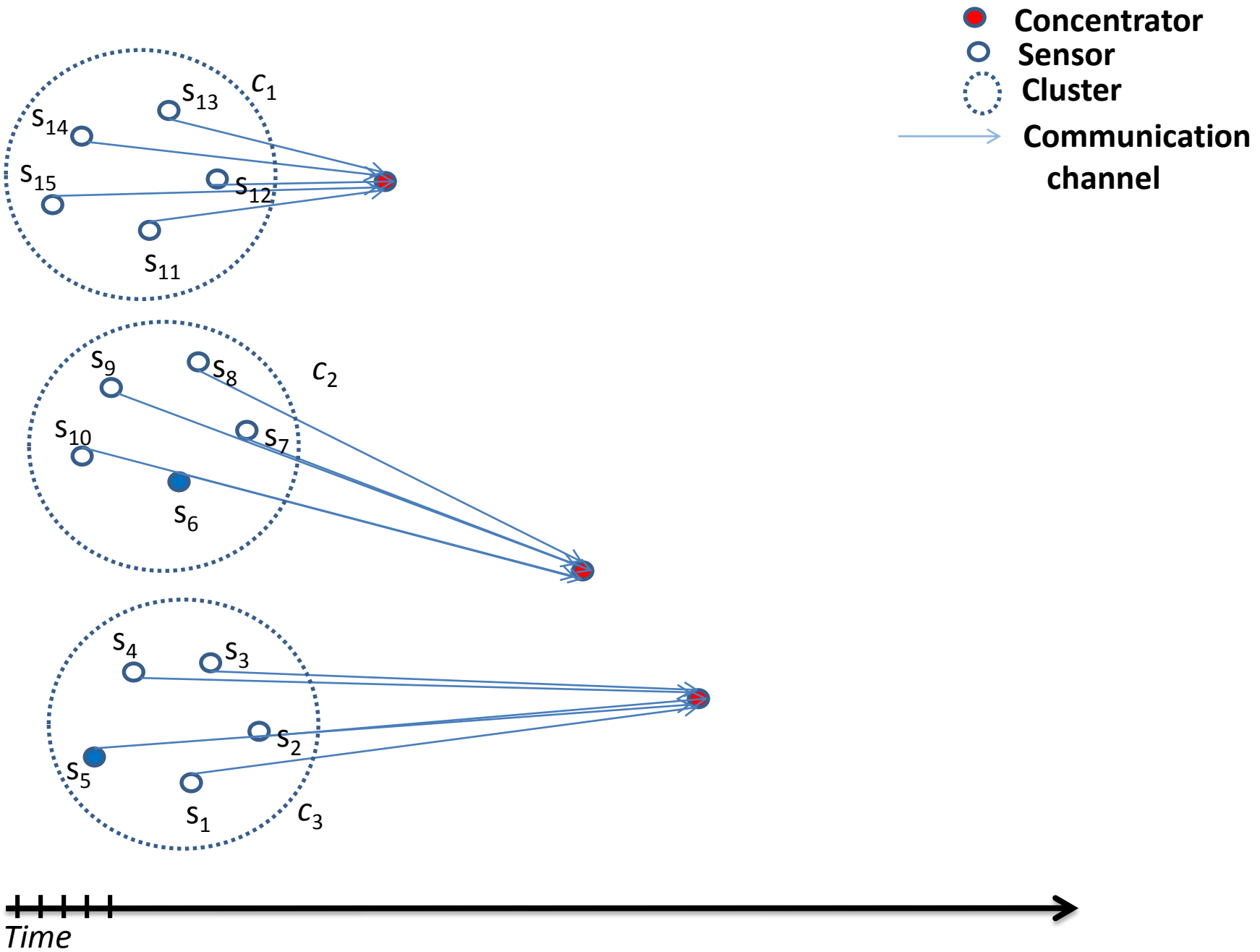
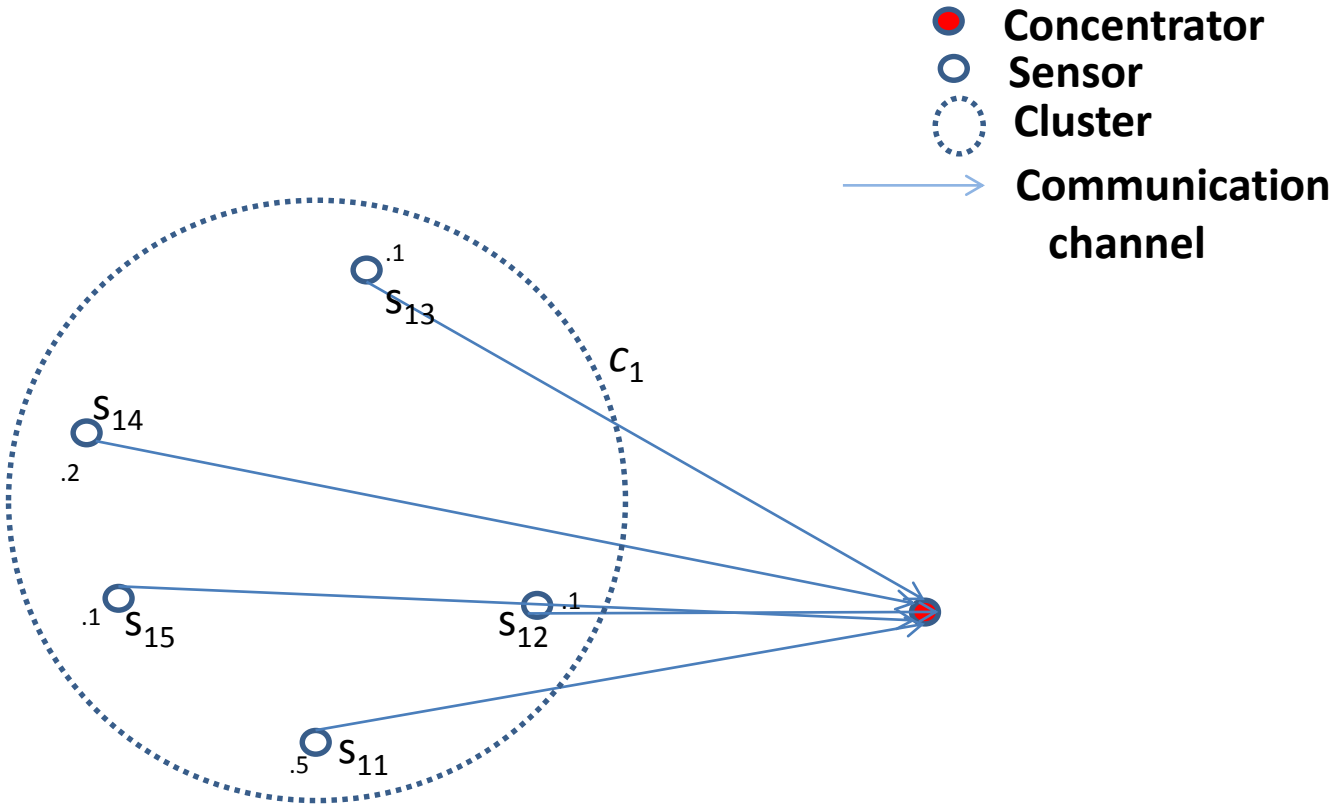


Figure 3b: Weights



Concentrator and Weight

1. A *concentrator* is a software implementation based on mathematical function(s) that transforms various sensor data into *intermediate* data.
2. A *distiller* is a software implementation based on mathematical function(s) that typically inputs intermediate data to produce more condensed intermediate
3. For each cluster there is one concentrator
4. A *weight* is the degree to which a particular sensor's data will impact a concentrator's computation. The concentrator will use weights to compute intermediate data
5. A weight can be hardwired or modified on the fly
6. A weight can be based on a sensor's perceived trustworthiness, e.g., based on who is the sensor's owner, manufacturer, geographic location of manufacture, geographic location where the sensor is operating, sensor age or version, previous failures or partial failures of sensor, sensor tampering, sensor delays in returning data, etc.
7. Concentrators may have intelligence and the ability to self-modify their abstract clusters as well as to modify weights
8. Concentrators may be acquired off-the-shelf
9. Security is a concern for concentrators (malware or general defects)
10. Reliability is a concern for concentrators (general defects).

Communication Channel

1. Communication channels move data between computing and sensing
2. Communication channels are shown as unidirectional in this simple model
3. Communication channels are often wireless
4. Communication channels are likely an offering (*service* or *product*) from 3rd party vendors
5. Communication channel *trustworthiness* affects the ability to move data
6. Communication channels are prone to disturbances, interruptions, and reduced reliability
7. *Redundancy* can improve communication channel reliability
8. Security is a concern for communication channels
9. Reliability is a concern for communication channels.

Figure 5: Concentrator Computation

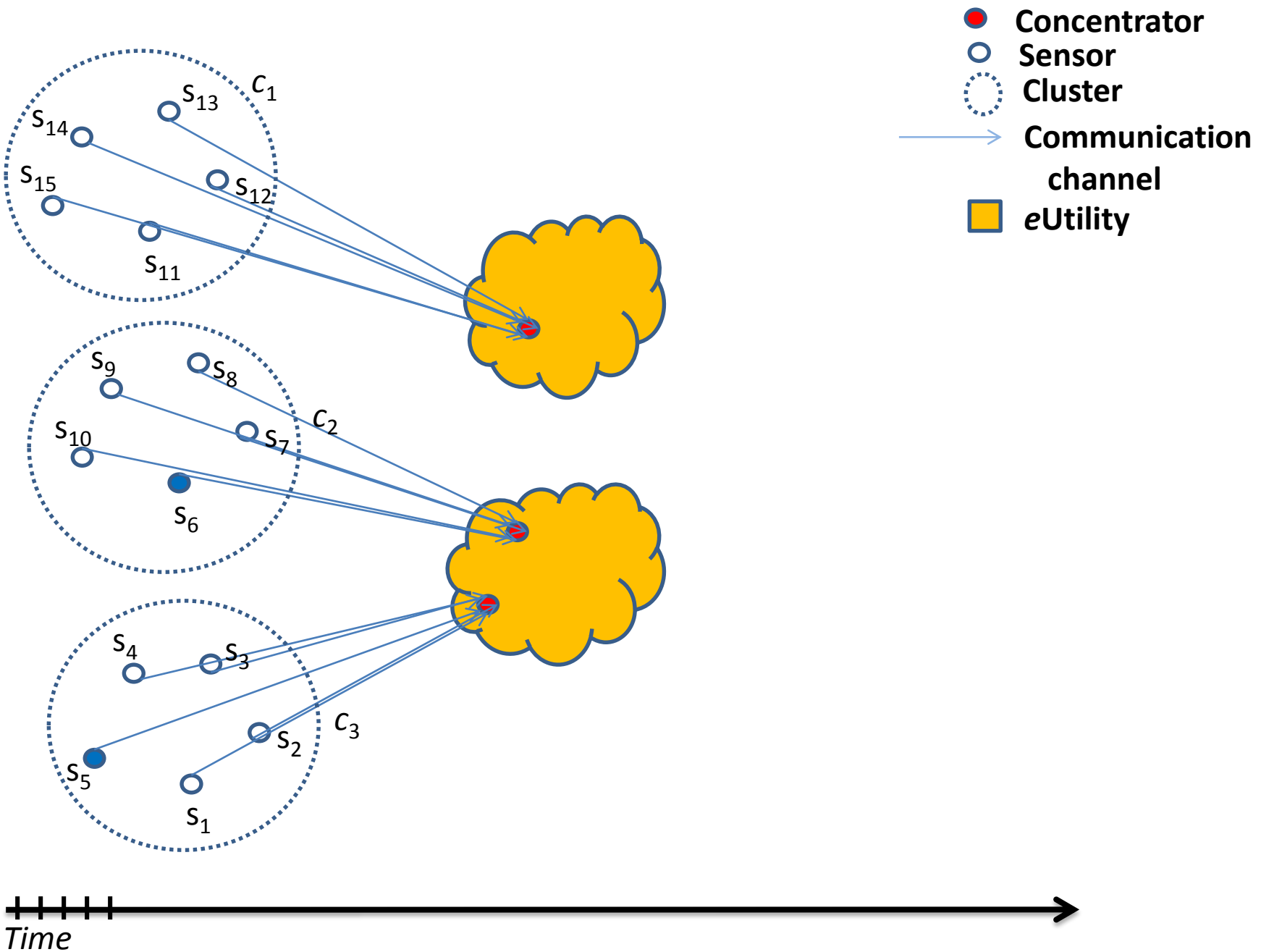


Figure 6: Distiller

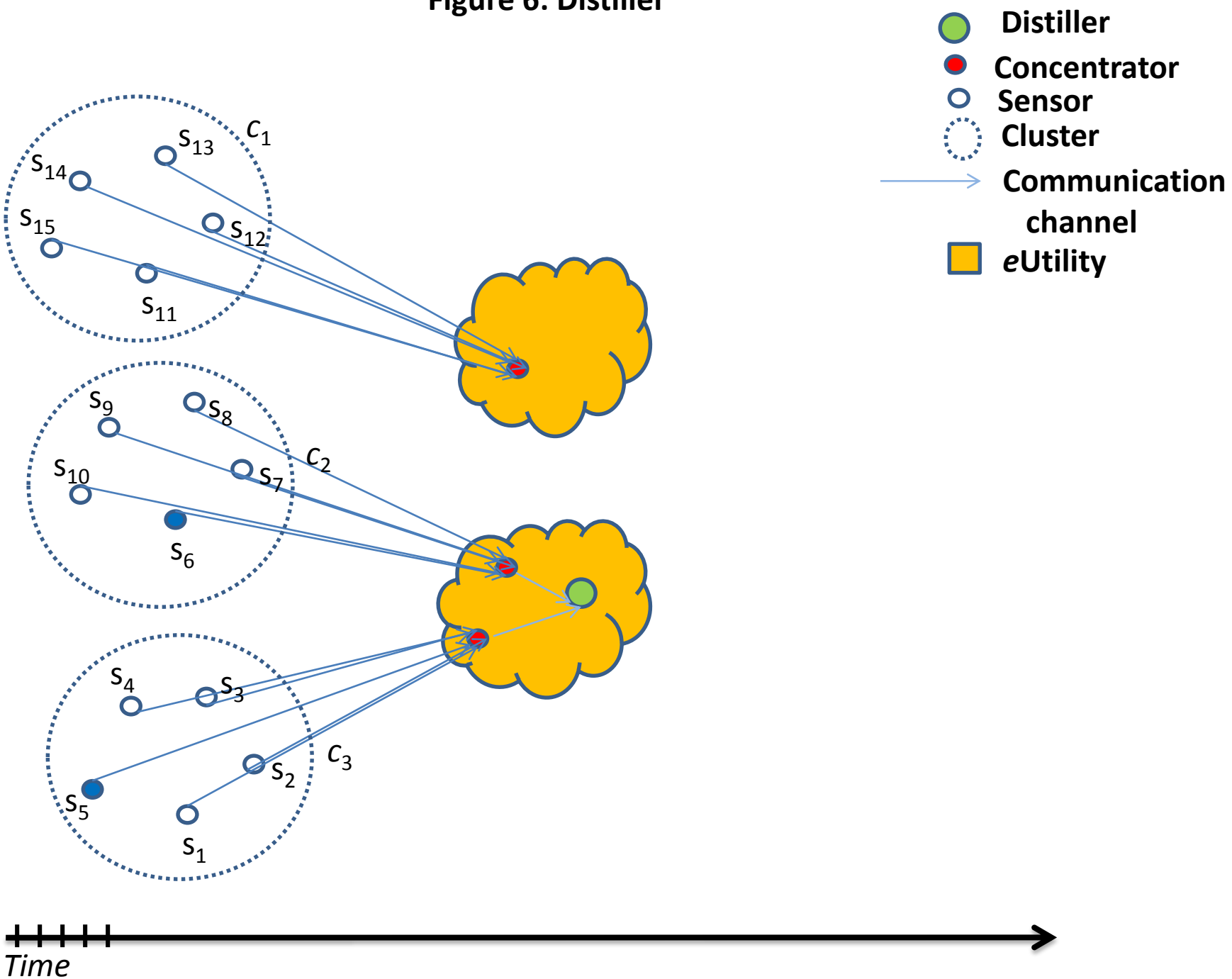
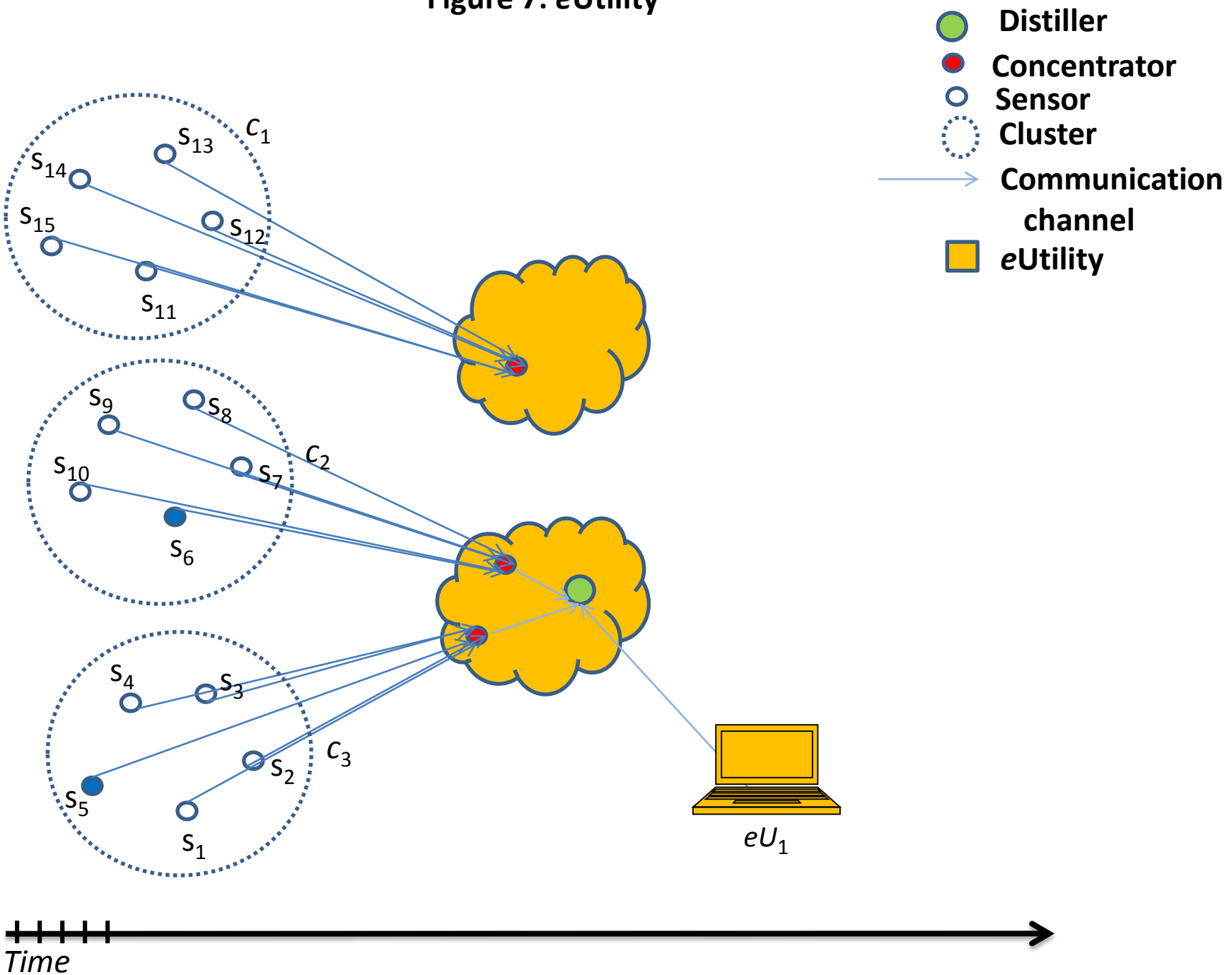


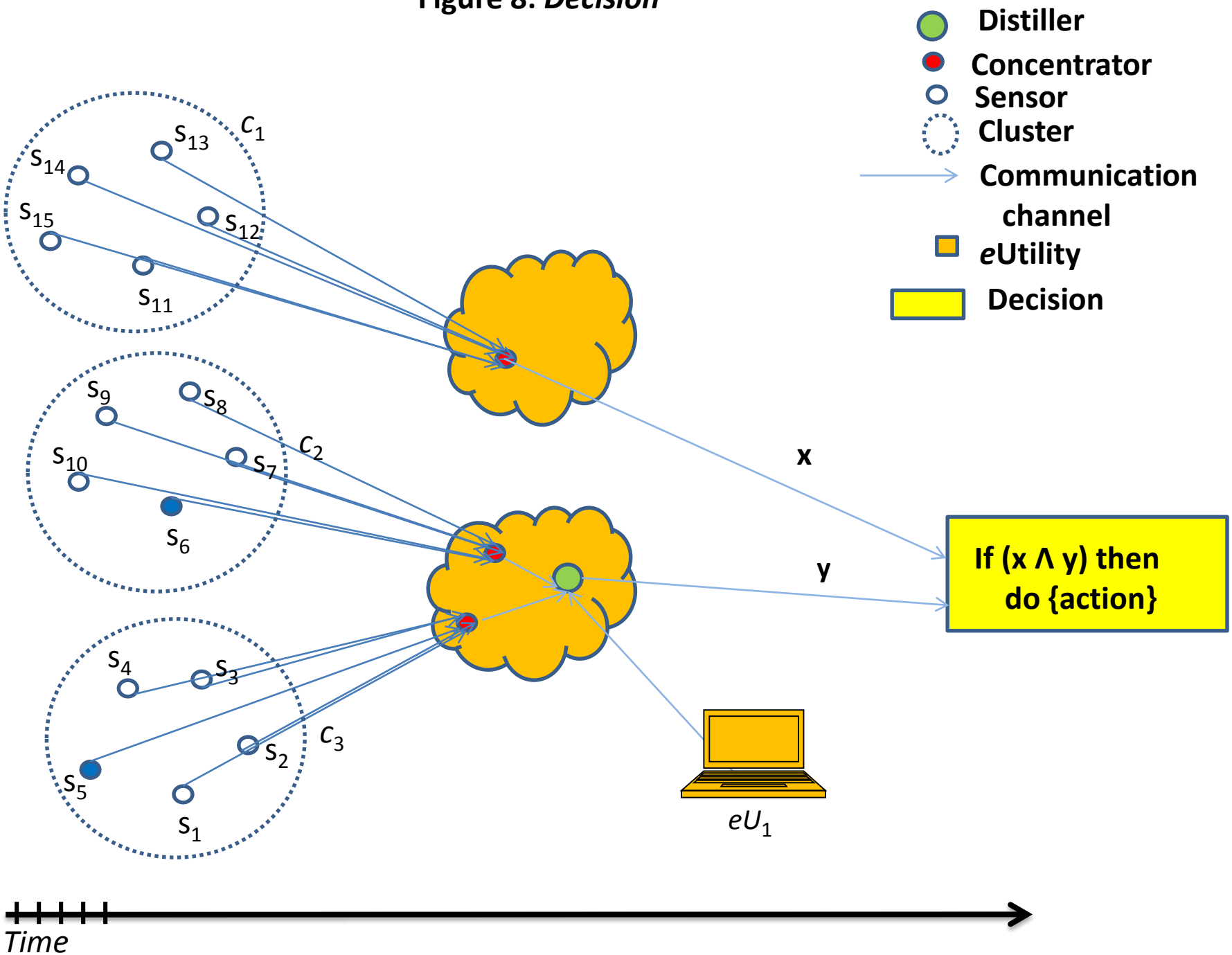
Figure 7: *e*Utility



*e*Utility

1. A software or hardware product or service that feeds data
2. *e*Utilities will likely be acquired off-the-shelf
3. *e*Utilities could databases, mobile devices, misc. software or hardware systems, clouds, computer, cpu, etc.
4. A human can be treated as an *e*Utility
5. Data supplied by an *e*Utility can be weighted
6. Security is a concern for *e*Utilities
7. Reliability is a concern for *e*Utilities.

Figure 8: *Decision*



Decision

1. A decision is the final result from data concentration, D = *f*(x, y) Decisions are the outputs of NoTs.
2. A decision has an unique owner
3. Decisions may be acquired off-the-shelf or homegrown
4. Decisions are made at a time snapshot and may occur continuously as new data becomes available
5. Decisions may be predictions
6. Decision results may control actuators or other transactions
7. The workflow from sensor data collection to decision making is partially parallelizable
8. Failure to make accurate decisions at time snapshot t_x may result because of tardy data collection, inhibited sensors or *e*Utilities, inhibited communication channels, and slow concentrators
9. Security is a concern for decisions (malware or general defects)
10. Reliability is a concern for decisions (general defects).

Ninth and Tenth NoT Primitives

1. Sensor
2. Time snapshot (time)
3. Cluster
4. Concentrator
5. Weight
6. Communication channel
7. *e*Utility
8. Decision
9. Geographic location
10. Owner

Geographic Location

1. Place where sensor or *e*Utility operates – these may change locations

2. Place where sensor or *e*Utility was manufactured

3. A sensor's geographic location along with its communication channel reliability may affect the ability to move data from sensor to concentrator.

Owner

1. Person or Organization that owns a particular sensor, communication channel, concentrator, decision, *e*Utility, or computing platform

2. Owners may have nefarious intentions.

Summary

1. A common vocabulary is useful to foster dialogue concerning IoT
2. 10 primitives that impact the trustworthiness of NoTs are proposed
3. NoTs or 'subnets of things' are the likely means by which IoT will be delivered
4. IoT is in part a *big data* problem
5. Of the 10 primitives, time snapshot is likely the most over-looked yet possibly the most important
6. Another consideration, *environment* (*context*), seems necessary, but for now is elusive
7. Another consideration, *cost*, can be added to turn this approach into an economic model by identifying risk trade-offs, e.g., the security, reliability, and performance associated with each of the 10 primitives "**The 'Internet of Things' Will Be The World's Most Massive Device Market And Save Companies Billions Of Dollars"** http://www.businessinsider.com/how-the-internet-of-things-market-will-grow-2014-10
8. Standards are needed at the implementation level "**Standards will be critical for the emergence of the Internet of Things"** http://m2mworldnews.com/2014/07/31/80797-standards-will-be-critical-for-the-emergence-of-the-internet-of-things/
9. December 25, 2014 was declared the unofficial IoT Day. Why?

Points to Ponder (Version 1)

1. Things may be all software or hardware, a combination, or human.
2. Things may have a stealth/invisible mode coming and going creating zero *traceability*.
3. Threats from previous genres of complex software-centric systems apply to NoTs . Security threats in NoTs may be exacerbated as a result of composing 3rd party things. This may create an *emergent* class of threats.
4. Functional composition ≠ Secure composition.
5. Forensics concerning security for billions of late-binding heterogeneous things is unrealistic.
6. Counterfeit things is a *supply chain* problem.
7. *Authentication* addresses the 'Who 's Who' and 'What's What' questions. Things may misidentify.
8. *Actuators* are things; if fed malicious data from 'other things', issues with life-threatening consequences are possible.
9. NoTs are *time*-sensitive. Defective local/global clocks (timing failures) lead to deadlock, race conditions, and other classes of system-wide failure.
10. Some NoTs may have the ability to self-organize and self-modify (self-repair). If true, NoTs can potentially rewire their security policy mechanisms/implementations or disengage them altogether.

Points to Ponder (Version 2)

1. Most known threats from previous genres of complex software-centric systems apply to NoTs. No new security threats are currently known (to us) to be exclusive to NoTs.
2. Functional composition ≠ Security composition.
3. Size (number of things) fuels unbounded complexity in functionality and diminishes any hope of *testability* via *observability* and instrumentation.
4. A Private Network of Things (PNoTs) may bound scalability and complexity, and if so they might enhance the argument for trustworthiness.
5. A PNoT may or may not use 'things' tied to the Internet.
6. Security flaws/threats in NoTs will be exacerbated by composition of 3rd party things. This creates an *emergent* class of security unknowns.
7. Some NoTs may have the ability to self-organize and self-modify (self-repair) if 'smart' and AI are introduced. If true, NoTs can potentially rewire their security policy mechanisms/implementations or disengage them altogether.
8. Forensics concerning security for billions of composed, heterogeneous things is not possible in linear time. Like NP-hard or NP-complete. Intractable.
9. Things will be heterogeneous, (e.g., manufacturer, complexity, functionality). Thing counterfeiting may lead to seemingly non-deterministic behavior making testing's results appear chaotic. Counterfeit things lead to illegitimate NoTs, a supply chain problem.
10. *Authentication* of sensors will be a principle data integrity risk – the 'Who is Who' question. Things may misidentify themselves.
11. Things may have a stealth/invisible mode coming and going (instantaneous snapshots now you see it now you don't), leaving zero *traceability*.
12. Things may be all software or hardware, a combination, or human.
13. Things will likely sense. Sensors will communicate. Sensors may compute but will be limited to very small code bases and CPUs. Security problems as a result of intercommunication (in general) will affect NoTs. *Wireless* is intercommunication example.
14. *Actuators* are things; if fed malicious data from 'other things', issues with life-threatening consequences are possible, an 'ility to ility' example of a '*shall not*' requirement.
15. NoTs are highly *time*-sensitive – NoTs need *synchronization*. Defective local/global clocks (timing failures) lead to deadlock, race conditions, and other classes of system-wide failure.