Link Aggregation Control Protocol

Presentation to the Link Aggregation Task Force, July 1998

Tony Jeffree

Alan Chambers

Overview

- Uses the best bits of the Finn/Wakerley/Fine & Jeffree presentations from the Interim meeting
- Much work done on the protocol description & operation
- Subdivision into more, simpler state machines for clarity & functional independence

Basic assumptions/objectives

- If aggregation is possible, it will happen automatically
- If not possible, links operate normally
- Determinism
- Rapid convergence
- Low risk of misconfiguration
- Low risk of duplication or misordering

Specific Objectives - 1

- Ability to configure “speak if spoken to” Ports (= Automatic mode) and “speak anyway” Ports (= Desirable mode)
- Ability to detect “crowds” - multiple partners on shared medium links
- BUT should not be necessary to switch the protocol off to get today’s performance on shared media links with h/w that cannot distinguish point-to-point vs shared

Specific Objectives - 2

- Ability to configure “Relaxed” operation for Ports that can hardware detect link failure, or “Nervous” operation for Ports that cannot
- Fast detection of presence/absence of partners on initialisation
- Accommodation of hardware that can control transmit/receive independently, and of hardware that cannot

Specific Objectives - 3

- Fast detection of cases where aggregation cannot occur => activate as individual link
- Ability to determine which physical Ports can/cannot aggregate with which Aggregate Ports
- Very low probability of misdelivery
- Low probability of loss
- Low probability of reporting good link with only partial connectivity

Identifying link characteristics

- Many characteristics that contribute
  - Standardised in .3: Link speed, duplex/non-duplex…etc
  - Other characteristics…e.g., administrative, non-standardised
- A Link is allocated a single *Capability* Identifier
- *Capability Group:* All Links in a system that share the same Capability ID
- Links that are not capable are in a Capability Group with one member
- Links can only aggregate with Aggregators that have the same Capability ID

Identifying Link Aggregation Groups

- System ID plus Capability provides a global identifier for a Capability Group
- The set of links in an aggregation are identified by concatenating the Capability Group identifiers at each end of the link
- Hence, for Systems S and T, who use C and D as the Capability ID for a set of aggregated links, the LAG ID would be {SC, TD}…(which is the same identifier as {TD, SC})

Detecting Aggregation possibility

- Aggregation possibility can be detected simply by exchanging global Capability Group Ids across a link; each system can then see whether any other Links exist with the same {SC,TD} value.
- If other links in a system exist with the same {SC, TD} then they can all be added to the same Link Aggregation Group
- Simplifying assumption: no limit on aggregation size - allocate more capabilities if it is necessary to impose such a limit.

Prevention of Duplication/Reordering

- Collect once you are in the right aggregation
- Don’t Distribute until you know that the other end is Collecting
- Stop Distribution/Collection on a Link prior to moving it to a new aggregation
- BUT also need to accommodate equipment which cannot switch collector/distributor independently
- Need to “flush” other links if Conversations are re-allocated as a result of adding/removing links

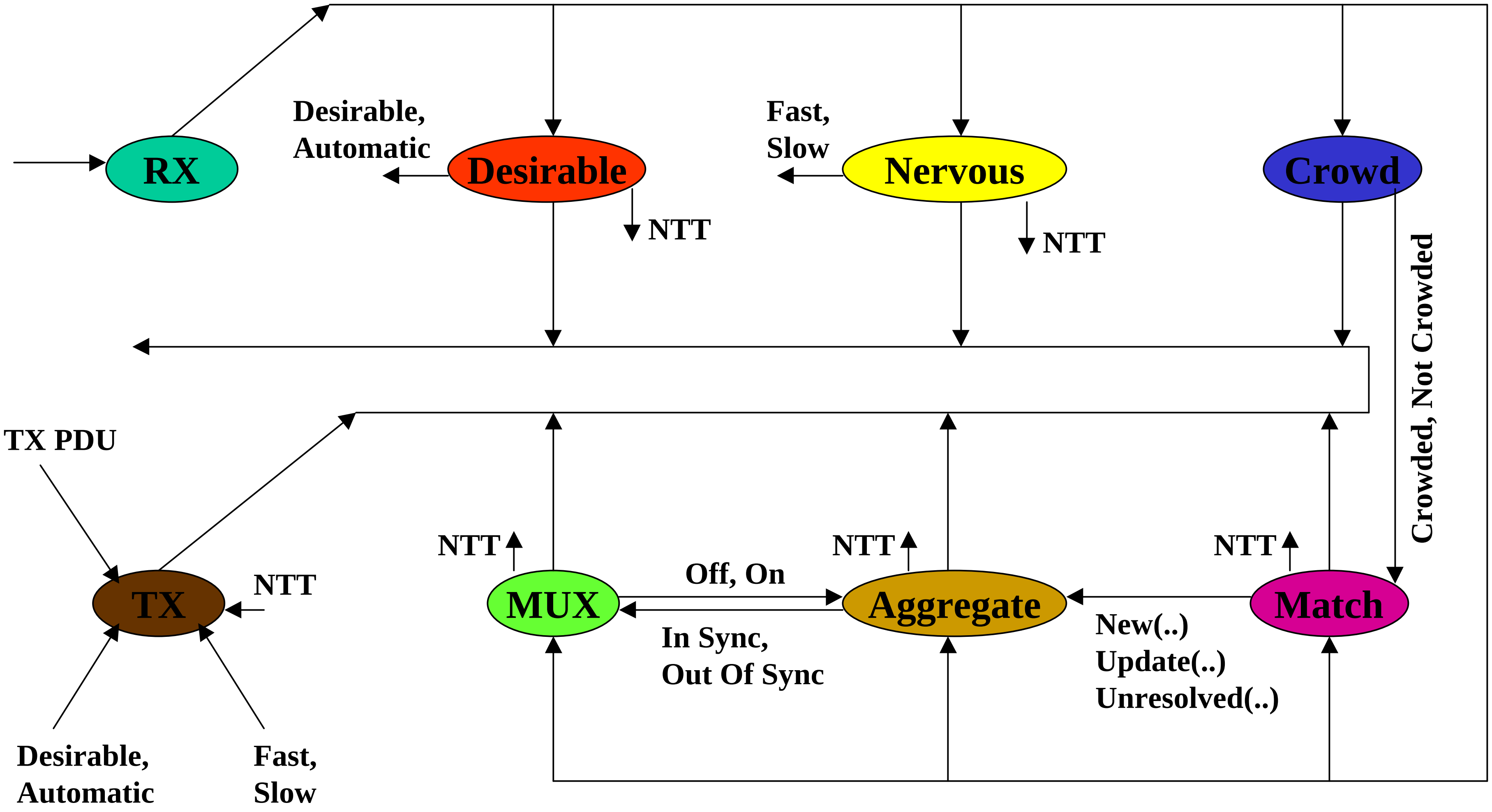
Protocol basics

- If the other guy doesn’t get it, say it again
- Assumption that packet loss is very low
- Communicate *state*, not *commands*
- *Need to Tell* if local state has changed, if information is old, or if the other guy does not get it
- Tell the other party what you know. When you are both agreed - aggregate

Flush protocol operation something like...

- Flush ID sent (along with normal message content). Sender chooses ID value.
- Recipient’s NTT is asserted by receipt of Flush ID; Flush ID saved by recipient & sent in subsequent messages till message received with no Flush ID.
- Note: Does not fix the case of a link failing.

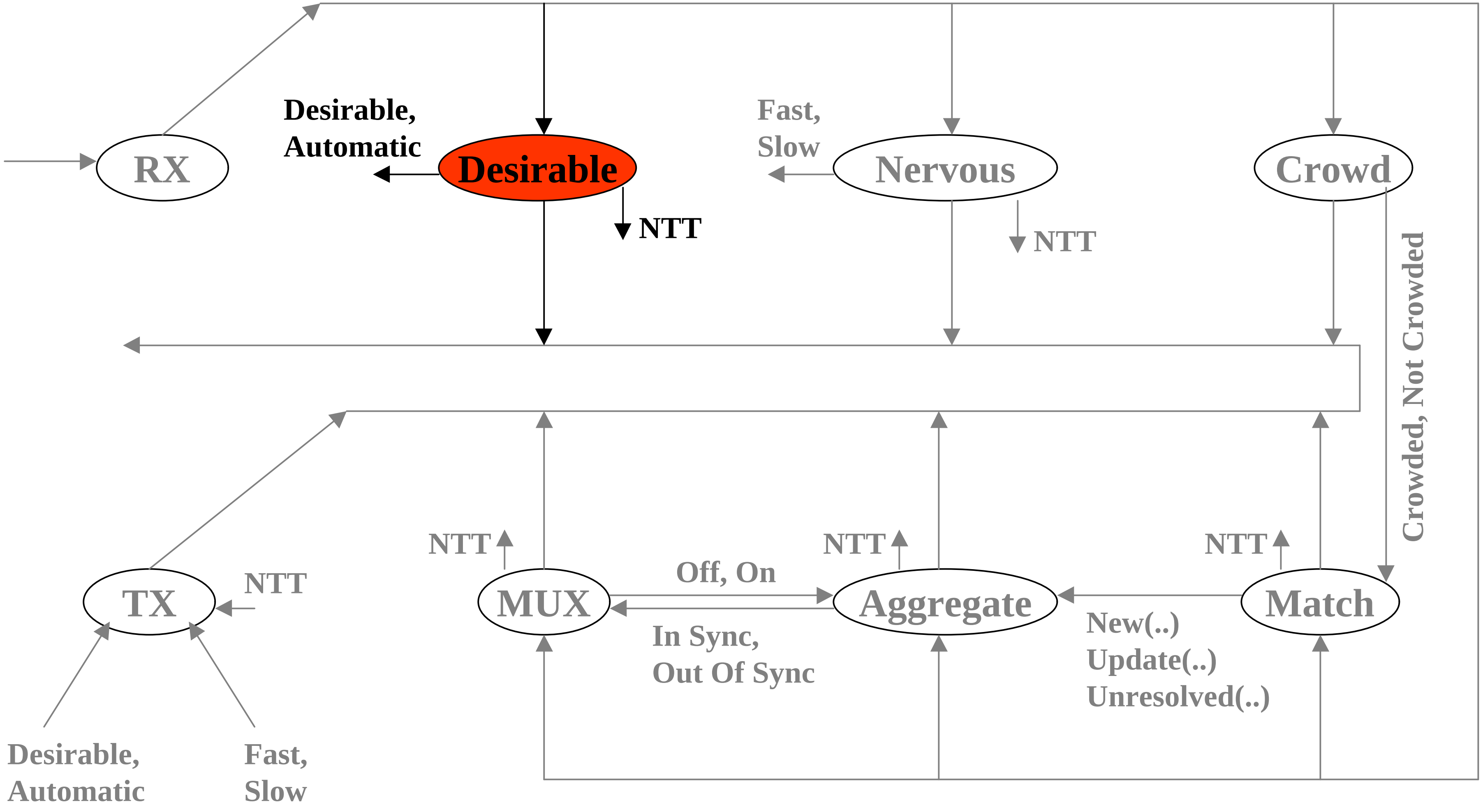
The Big Picture



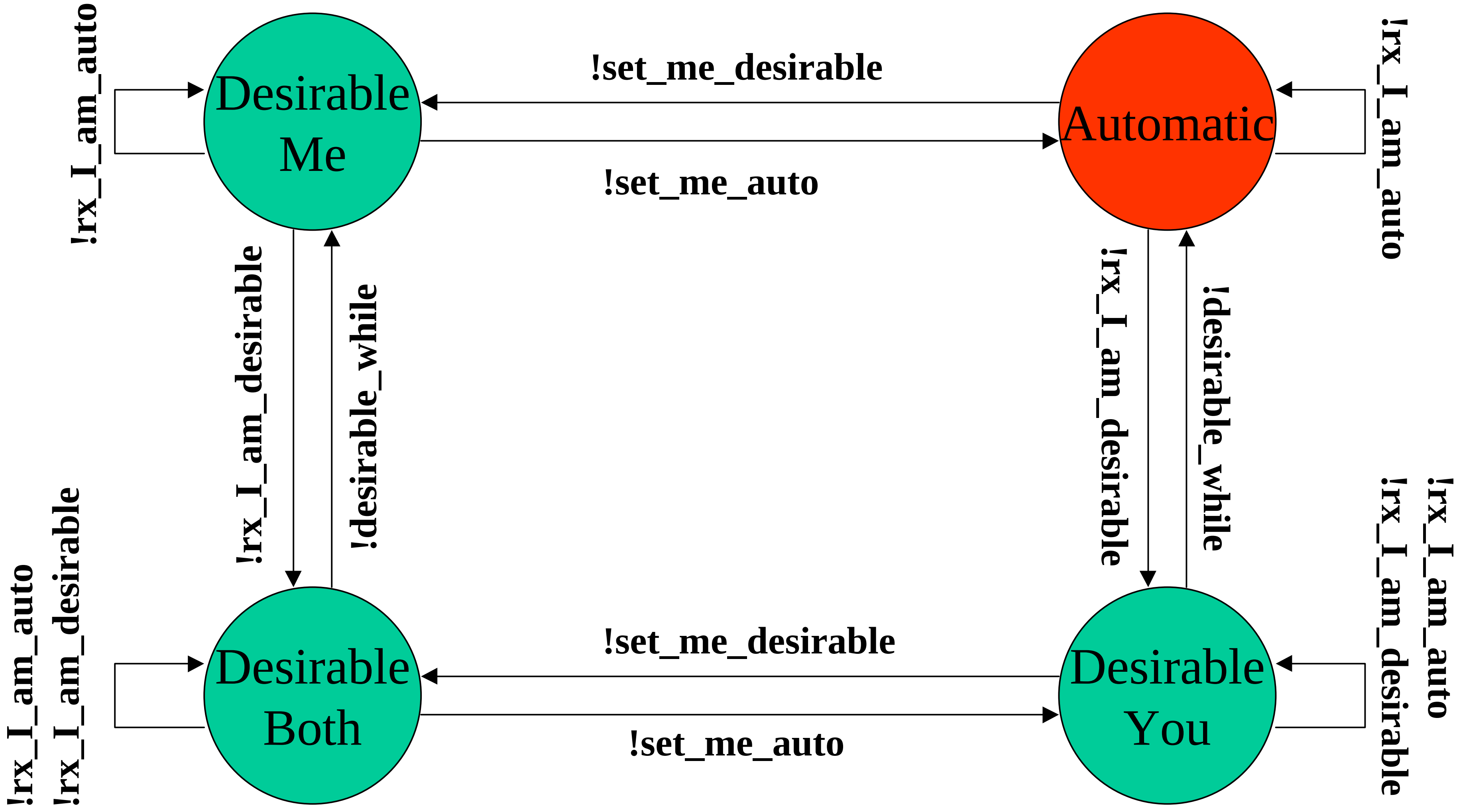
Information communicated

- My Port
- I Am Desirable
- Partner Desirable
- I Am Nervous
- Partner Nervous
- I Am Crowded
- Partner Crowded
- I Am Individual
- Partner Individual
- Sync
- I Am Collecting
- I Am Distributing
- Partner Collecting
- Partner Distributing
- My System
- My Capability
- Partner System
- Partner Capability

Desirable


RX
Desirable,
Automatic
Desirable
NTT
Fast,
Slow
Nervous
NTT
Crowd
Crowded, Not Crowded
NTT
NTT
NTT
NTT
TX
MUX
Off, On
In Sync,
Out Of Sync
Aggregate
New(..)
Update(..)
Unresolved(..)
Match
Desirable,
Automatic
Fast,
Slow

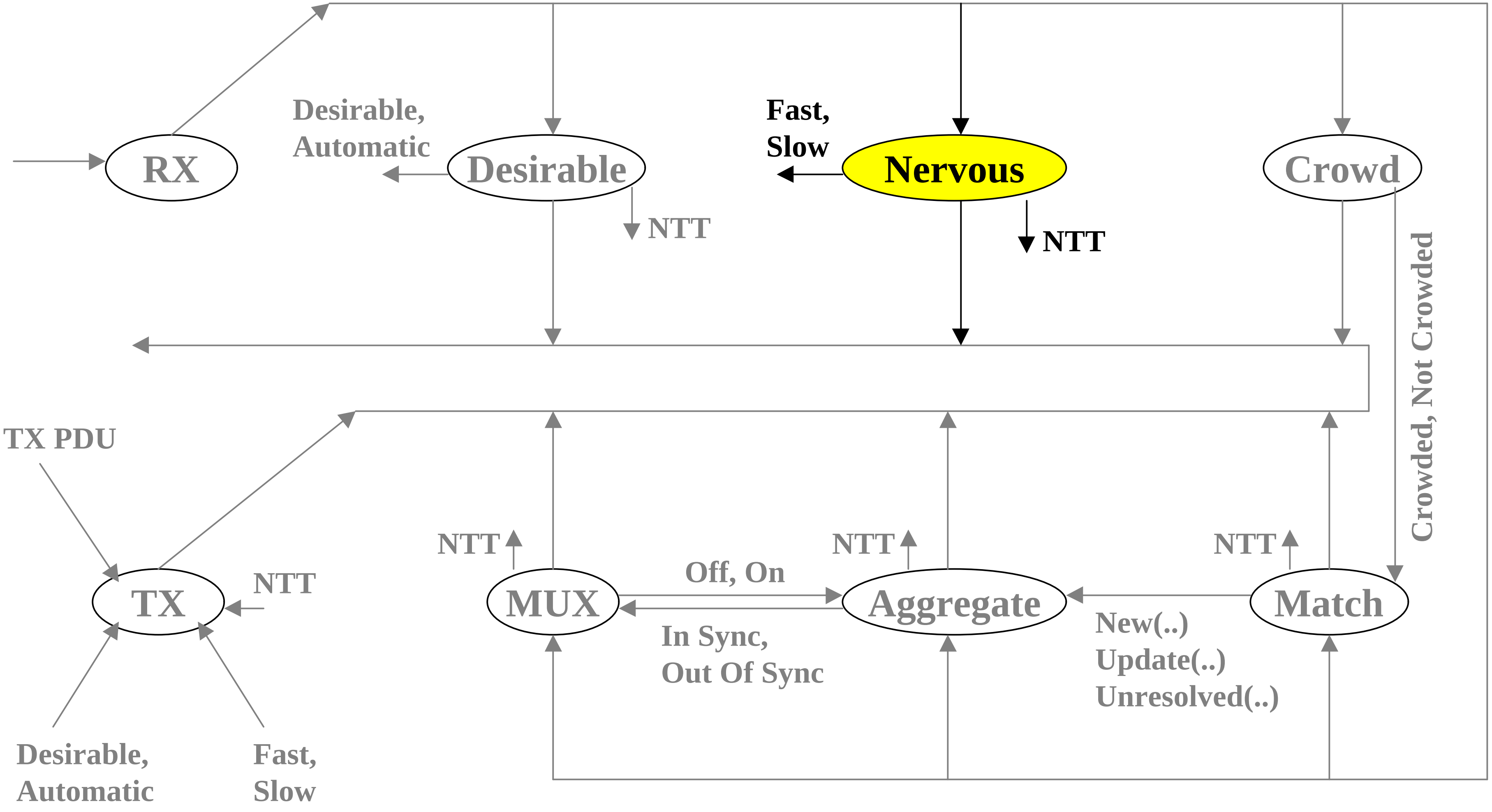
Desirable - State Machine



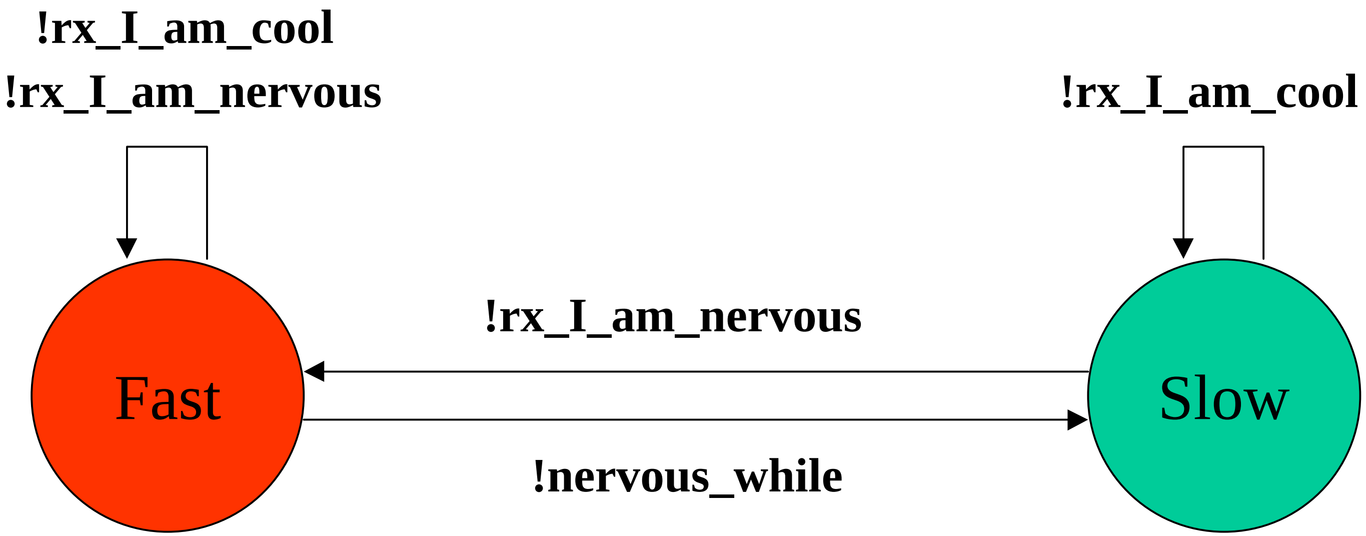
Desirable - Functionality Recap

- Determines whether or not this Port will generate routine LACPDUs
- *Desirable* if the actor or any of its partners are (or are believed to be) desirable
- *Automatic* if the actor and all of its partners are (or are believed to be) automatic
- If *automatic* this must be an individual link
- NTT if he doesn’t know my state
- Initial state: Partner is desirable

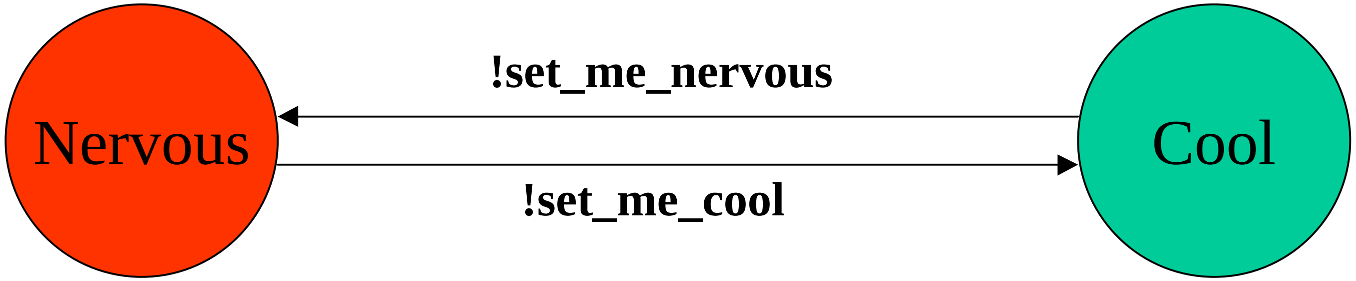
Nervous


RX
Desirable,
Automatic
Desirable
NTT
Fast,
Slow
Nervous
NTT
Crowd
Crowded, Not Crowded
TX PDU
TX
NTT
Desirable,
Automatic
Fast,
Slow
NTT
MUX
Off, On
In Sync,
Out Of Sync
NTT
Aggregate
NTT
Match
New(..)
Update(..)
Unresolved(..)

Nervous -
His Anxiety State Machine



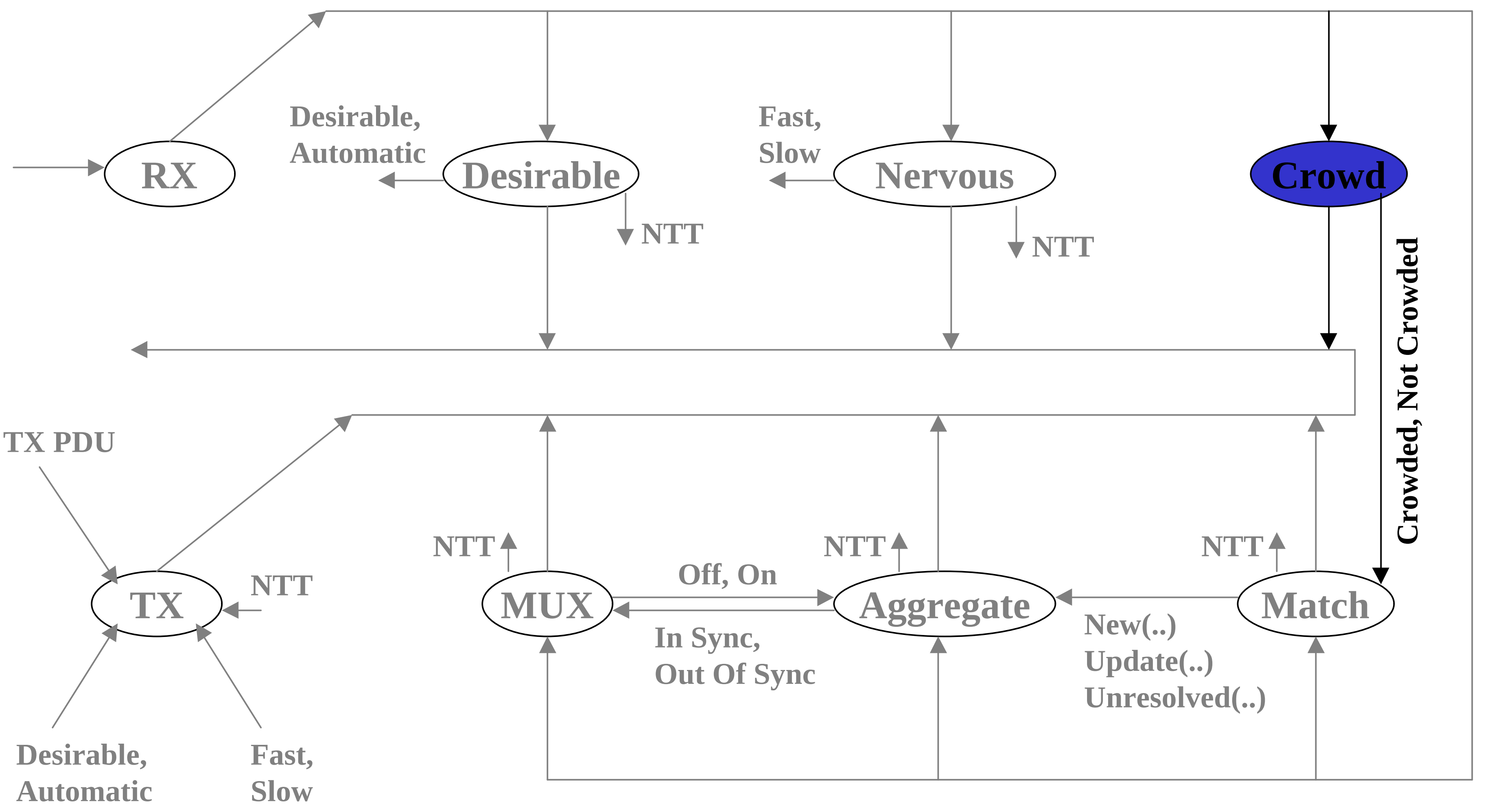
Nervous -
My Anxiety State Machine



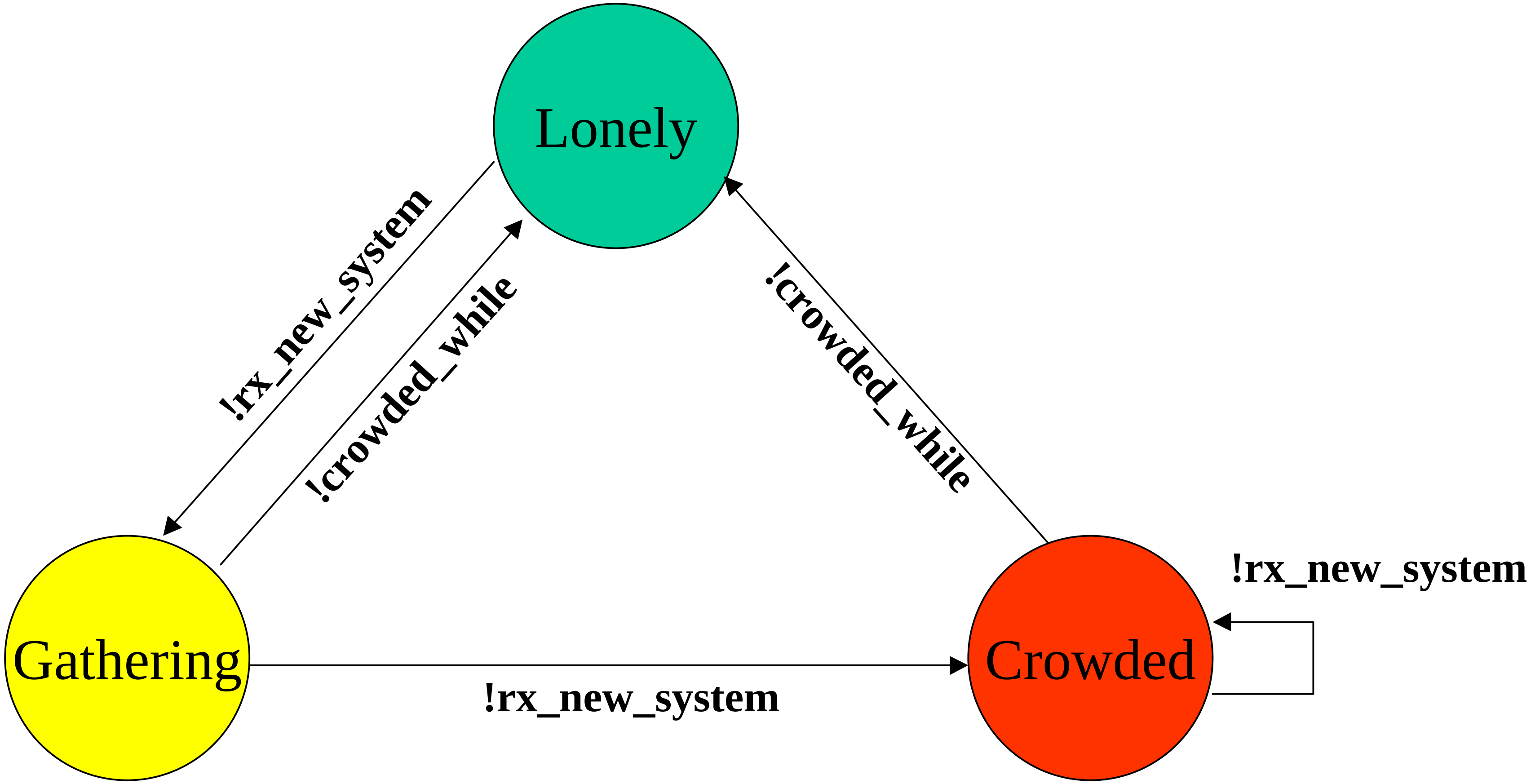
Nervous - Functionality Recap

- Controls whether routine LACPDU transmission is fast or slow
- Speed depends upon the nervous condition of the partner(s), not the actor
- Initial state: Partner is nervous

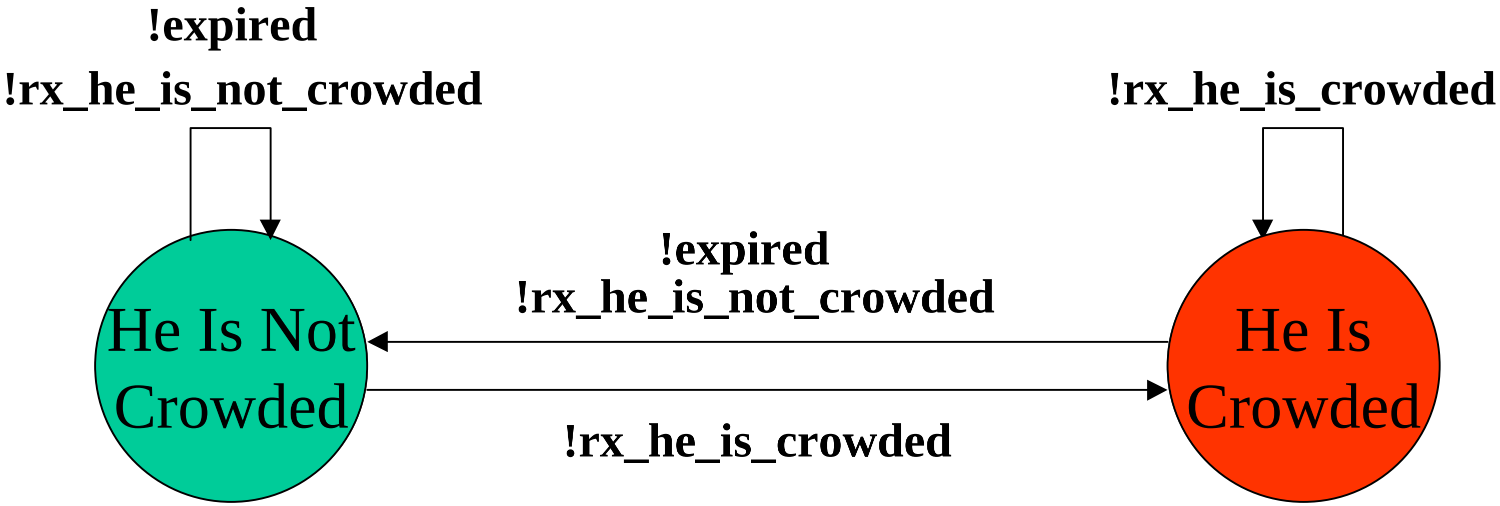
Crowd


RX
Desirable, Automatic
Desirable
NTT
Fast, Slow
Nervous
NTT
Crowd
Crowded, Not Crowded
TX PDU
TX
NTT
Desirable, Automatic
Fast, Slow
NTT
MUX
Off, On
In Sync, Out Of Sync
NTT
Aggregate
NTT
Match
New(..)
Update(..)
Unresolved(..)

Crowd -
Crowd Control State Machine



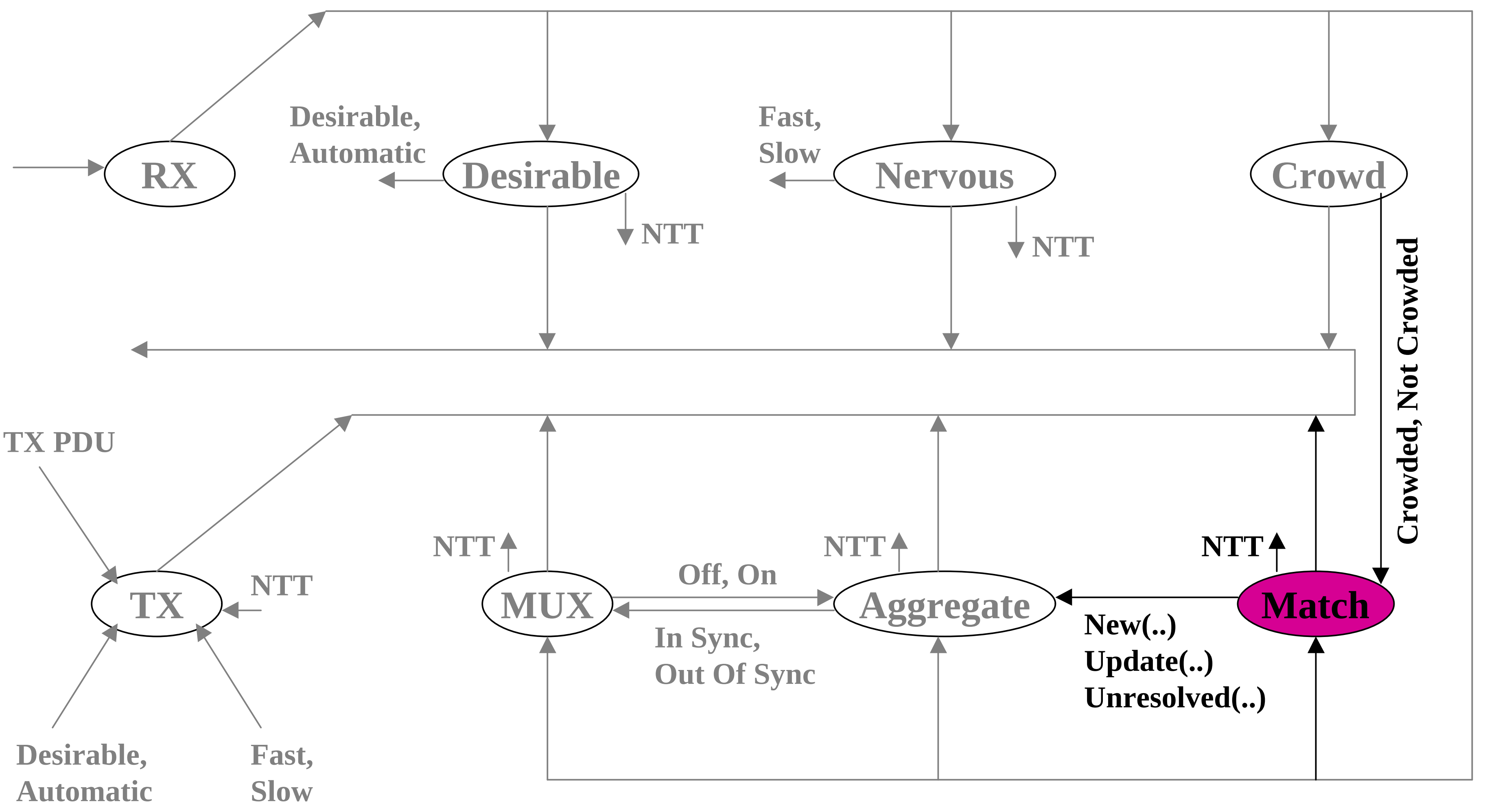
Crowd -
Crowd Records State Machine



Crowd - Functionality Recap

- Detects the presence of a *crowd* on the link
  - N’s company, N+1’s a crowd
  - N = 2 but model can be generalised
- I think there is a crowd if my N is exceeded
- There is a crowd if any partner’s N is exceeded
- Crowded links can only be individual links
- Gathering state avoids gratuitous disruption
- Initial state: Lonely

Match


RX
Desirable,
Automatic
Desirable
NTT
Fast,
Slow
Nervous
NTT
Crowd
Crowded, Not Crowded
TX PDU
TX
NTT
Desirable,
Automatic
Fast,
Slow
NTT
MUX
Off, On
In Sync,
Out Of Sync
NTT
Aggregate
NTT
Match
New(..)
Update(..)
Unresolved(..)

Match - State Machine (1)

- Have we agreed capabilities? We have if:
  - He is not crowded, we are not crowded, and he does not know of anyone who is crowded, and he has correctly identified our system and capabilities; or
  - He is crowded (that’s his decision, we have to agree); or
  - We are crowded, and he has agreed.

Match - State Machine (2)

- Have we agreed that this physical Port cannot be aggregated with any other? We have an “agreed individual” if:
  - His system/capability is particular to this Port (that’s his decision, we have to agree); or
  - Our system/capability is particular to this Port , and he has agreed; or
  - He is crowded, so this is an individual link; or
  - We are crowded, and he has agreed; or
  - We are both Automatic

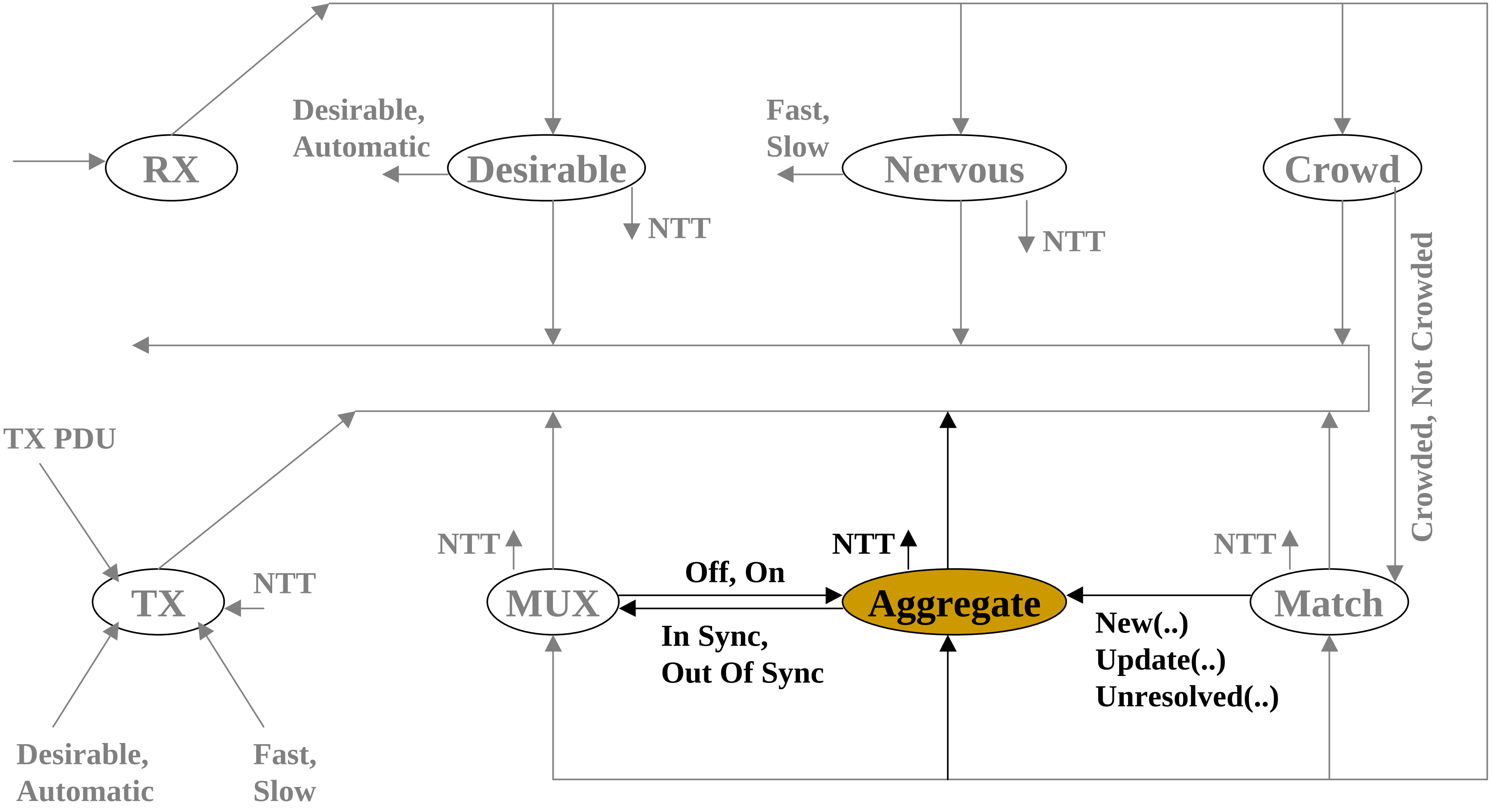
Match - State Machine (3)

- If we have not detected any partner on the link, then we are agreed (by definition, as we only have ourselves to agree with), and this link must be an individual link.
- If we have not reached agreement, NTT.

Match - Functionality Recap

- Determines whether or not the actor and its partner(s) agree on how the link should be aggregated
- Monitors and maintains the state of agreement
- NTT if no agreement reached
- Signals *new* aggregations, *updates* to aggregations, *unresolved* aggregations
- Initial state: No match

Aggregate


RX
Desirable,
Automatic
Desirable
NTT
Fast,
Slow
Nervous
NTT
Crowd
Crowded, Not Crowded
TX PDU
TX
NTT
Desirable,
Automatic
Fast,
Slow
NTT
MUX
Off, On
In Sync,
Out Of Sync
NTT
Aggregate
New(..)
Update(..)
Unresolved(..)
NTT
Match

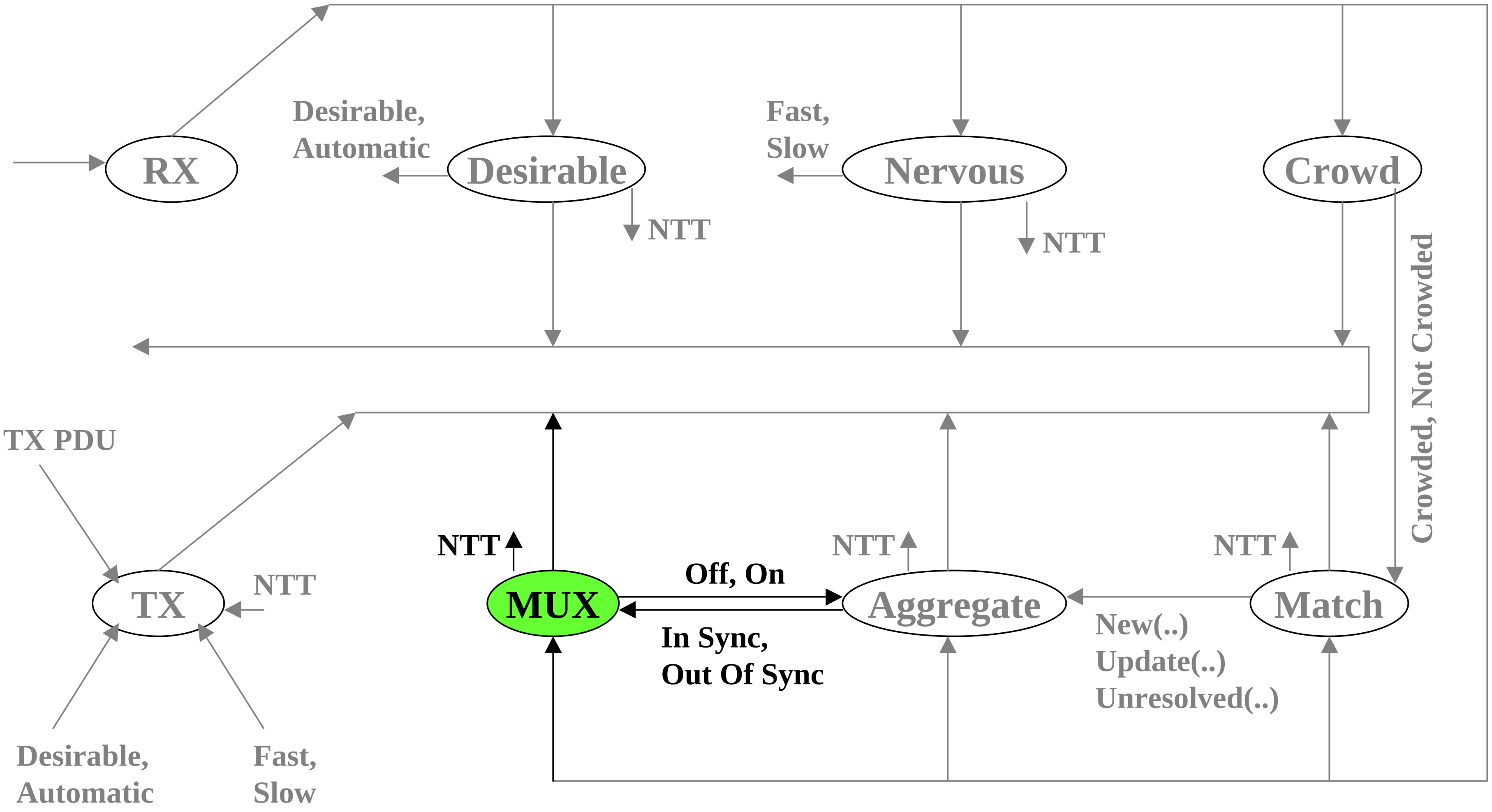
Aggregate - State Machine

- Once we have a Match, aggregates this physical Port with other compatible, matching Ports and a compatible Aggregate Port
- Deals with temporary resource shortages & delays

Aggregate - Functionality Recap

- Determines whether the link is in the right aggregate or not
- If not in the right one, removes it
- If not in an aggregate, finds the right one for it to be in and adds it
- Signals *in synch* when aggregated, *out of synch* when not
- Initial state: out of synch

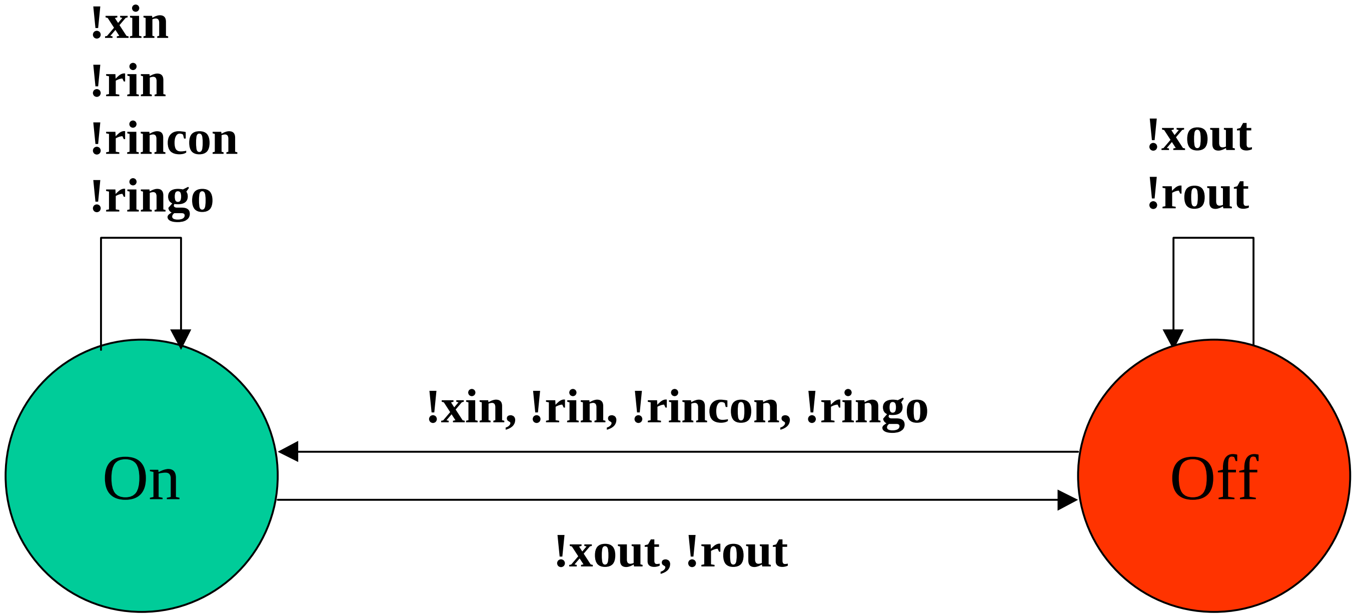
Mux


RX
Desirable,
Automatic
Desirable
NTT
Fast,
Slow
Nervous
NTT
Crowd
Crowded, Not Crowded
TX PDU
TX
NTT
Desirable,
Automatic
Fast,
Slow
NTT
MUX
Off, On
In Sync,
Out Of Sync
NTT
Aggregate
NTT
Match
New(..)
Update(..)
Unresolved(..)

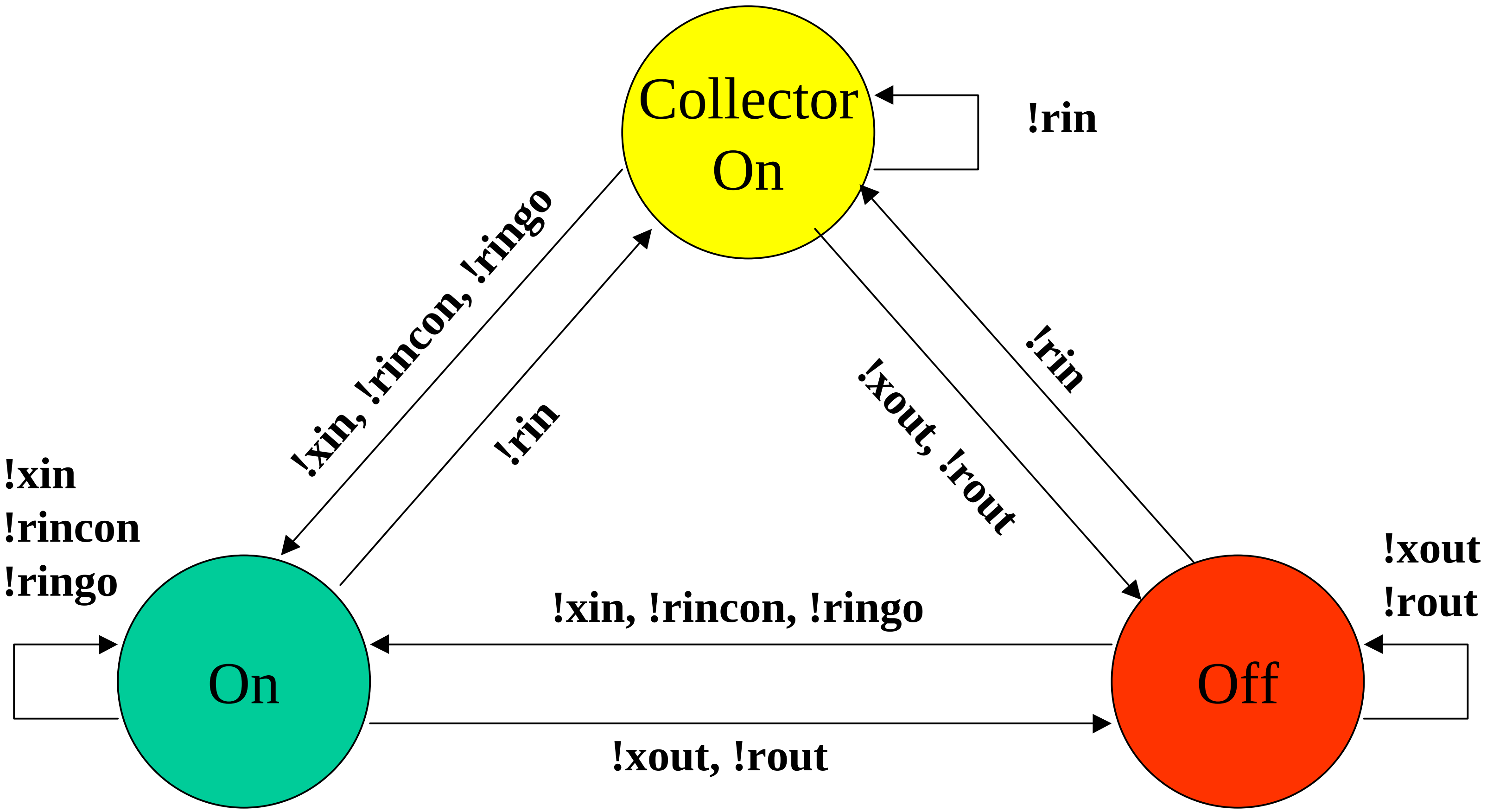
Mux - State Machine Events

- !xout - info expired, out of sync
- !xin - info expired, in sync
- !rout - received pdu, actor or partner out of sync
- !rin - received pdu, a & p in sync
- !rincon - received pdu, a & p in sync, p’s collector enabled
- !ringo - reeived pdu - both collector & distributor enabled
- !cop - from hardware, collector operational
- !dop - from hardware, distributor operational
- !cno - from hardware, collector not operational
- !dno - from hardware, distributor not operational
- !hop - from hardware, collector & distributor operational
- !hno - from hardware, collector & distributor not operational

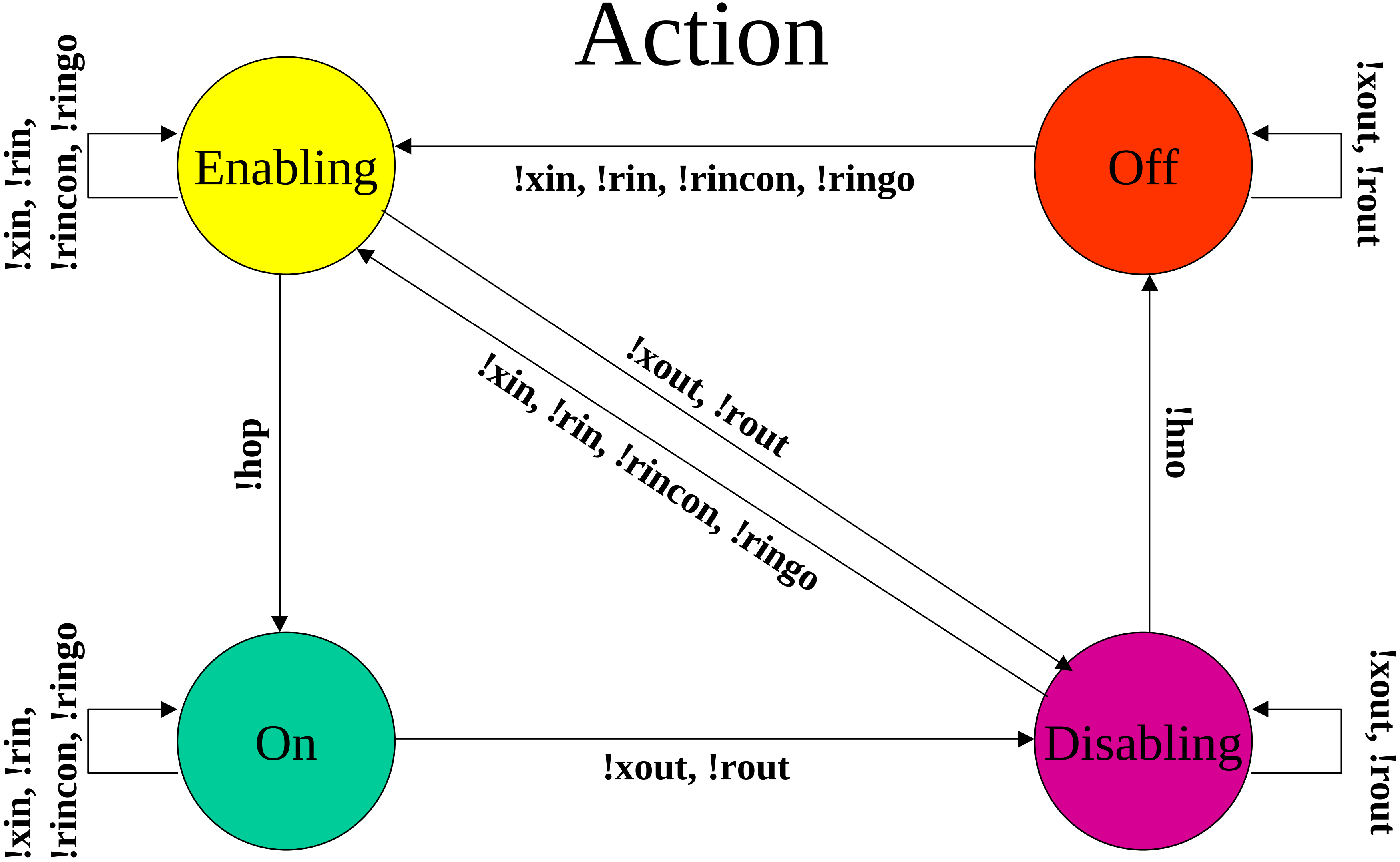
Mux - Coupled H/W, Immediate Action



Mux - Independent H/W, Immediate Action



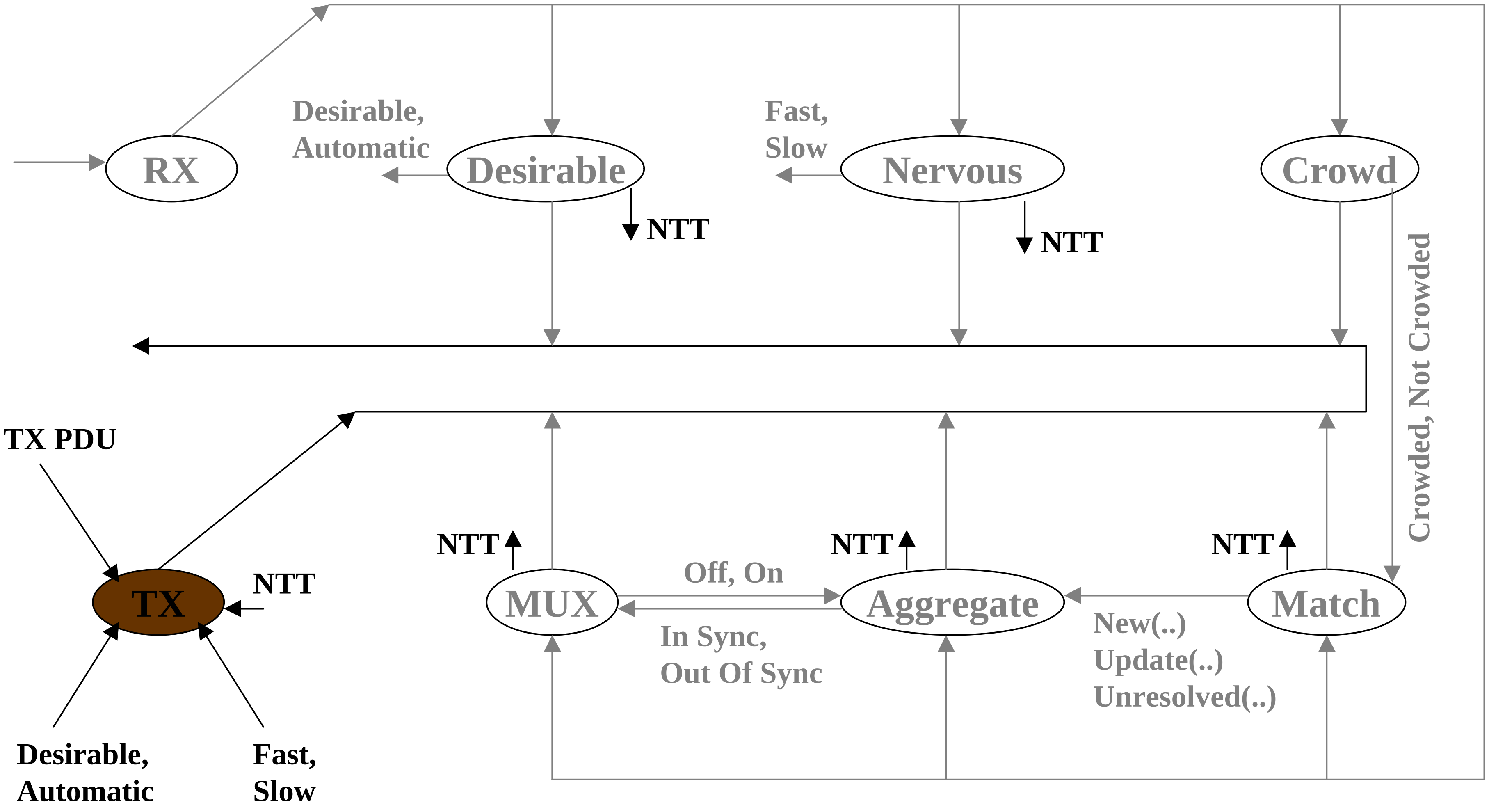
Mux - Coupled H/W, Delayed Action


Enabling
Off
On
Disabling
!xin, !rin, !rincon, !ringo
!xin, !rin, !rincon, !ringo
!xout, !rout
!xout, !rout
!xin, !rin, !rincon, !ringo
!xin, !rin, !rincon, !ringo
!xout, !rout
!xout, !rout
!hop
!hno

Mux - Functionality Recap

- When *in synch*, takes the necessary steps to turn on collector and distributor
- When *out of synch*, takes the necessary steps to turn off collector and distributor
- Signals *on*, *off* when its done
- Initial state: off

TX


RX
Desirable, Automatic
Desirable
NTT
Fast, Slow
Nervous
NTT
Crowd
Crowded, Not Crowded
TX PDU
TX
NTT
Desirable, Automatic
Fast, Slow
NTT
MUX
Off, On
In Sync, Out Of Sync
NTT
Aggregate
NTT
Match
New(..)
Update(..)
Unresolved(..)

TX - State Machine

	---	--t	-s-	-st	d--	d-t	ds-	dst
!ntt	--t	--t	-st	-st	d-t	d-t	dst	dst
Idesirable	d--	d-t	ds-	dst	d--	d-t	ds-	dst
!auto	---	--t	-s-	-st	---	--t	-s-	-st
!fast	---	--t	---	--t	d--	d-t	d--	d-t
!slow	-s-	-st	-s-	-st	ds-	dst	ds-	dst
!tx_when	---	--t	-s-	-st	d-t	d-t	dst	dst
	tx_when=F	tx_when=F	tx_when=S	tx_when=S	tx_when=F	tx_when=F	tx_when=S	tx_when=S
!tx pdu	---	---	-s-	-s-	d--	d--	ds-	ds-
		transmit		transmit		transmit		transmit

- State variables:

t = TX Pending/TX Not Pending

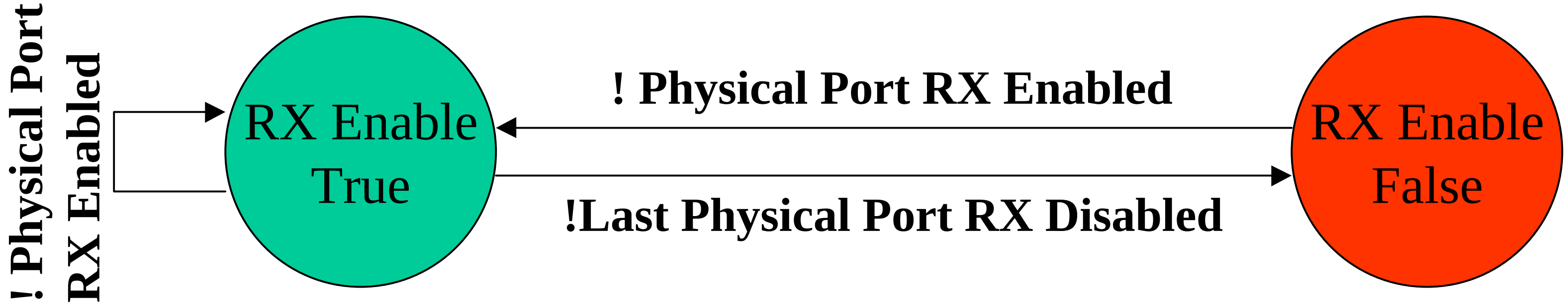
s = Slow/Fast

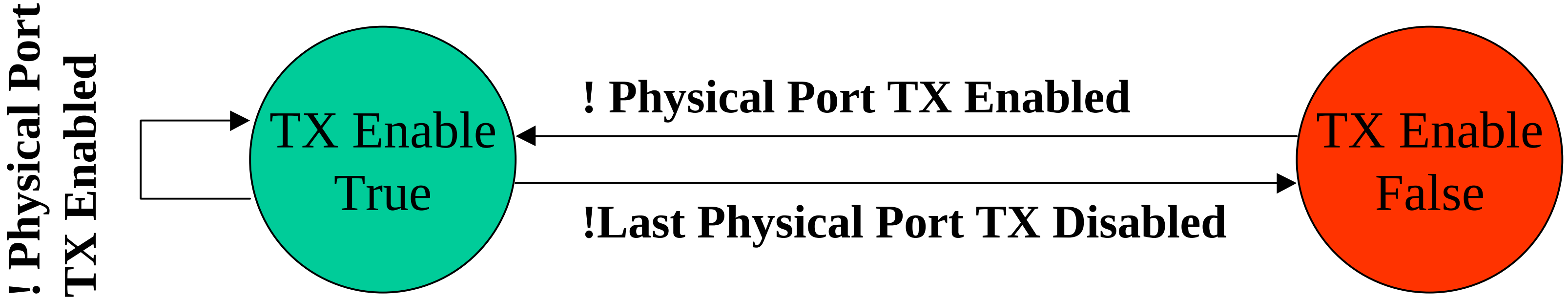
d = Desirable/Automatic

TX - Functionality Recap

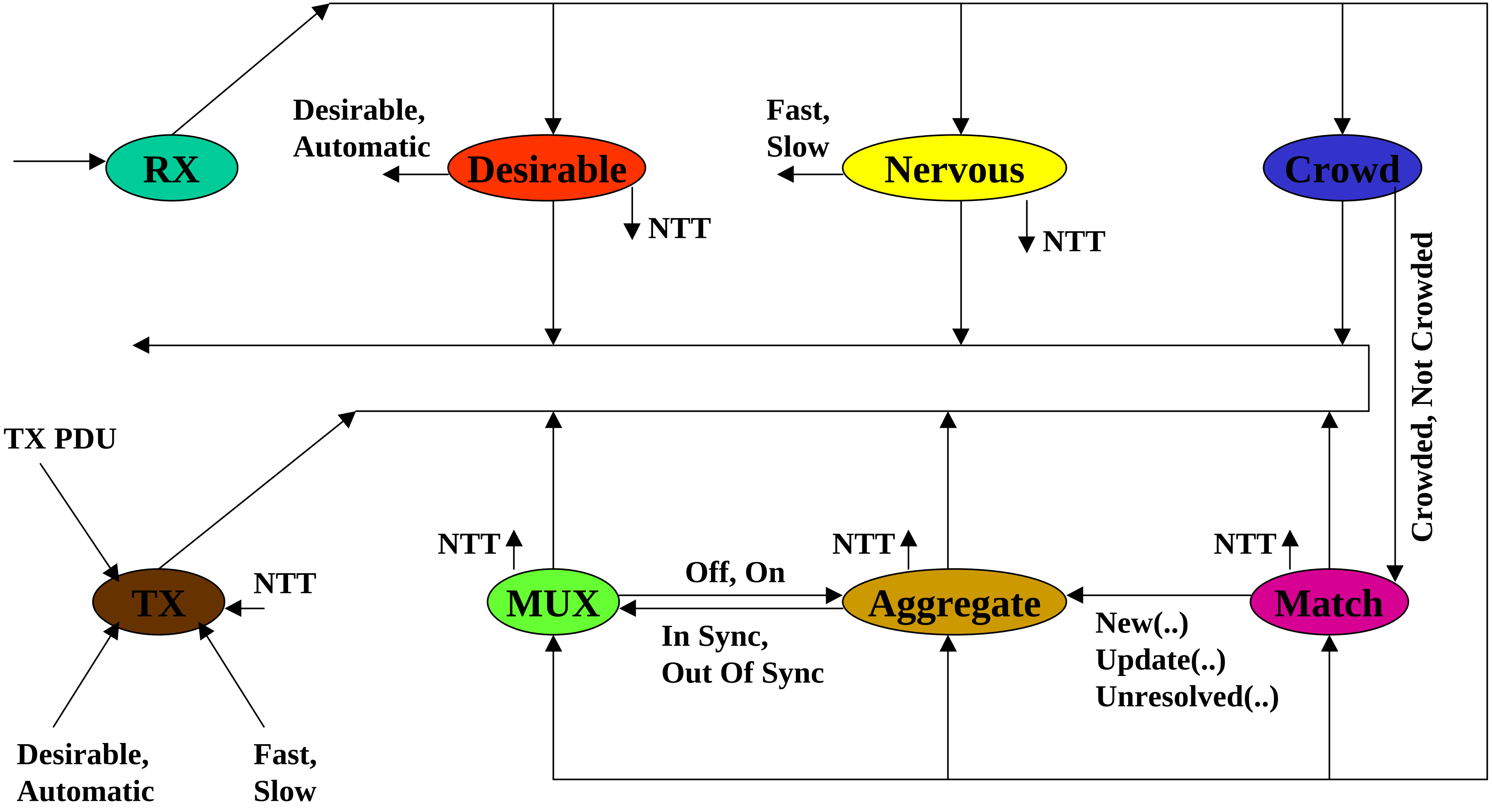
- Causes LACPDUs to be generated if:
- NTT
- Desirable
  - Frequency depends on *fast* or *slow*signal from Nervous state machine

Aggregate Port - State Machines





The Big Picture



Summary

- Covers (majority of) functionality described by Finn/Wakerly/Fine & Jeffree
- Fully describes the process of reaching agreement & the actions taken to join & leave aggregations
- Separate state machines improve clarity
- Flush protocol yet to be included

Example Protocol Scenarios

Link Configurations

X:/Y:	***APn***	***A0n***	***0Pn***	***00n***	***L***
APn	A1	A2	I1	I2	F1
A0n		(A3)	I3	(I4)	F2
0Pn			I5	I6	F3
00n				I7	F4

- A/0: Aggregatable/individual
- P/0: Preferred (Desirable)/automatic
- n: Nervous
- L: Legacy (non-participating device)
- Ai: Aggregate configurations
- Ii: Individual configurations
- Fi: Fall-back configurations, remote is a legacy non-participant

Notation for Examples

1st System		*2nd System*
X	System ID	Y
I	Capability ID	J
A	Aggregatable/Individual	B
P	Preferred {Desirable}/Auto mode	Q
N	Nervous/Relaxed	O
U	Uncrowded/Crowded	V
S	Sync/Out of Sync	T
C	Collecting	E
D	Distributing	F

Individual Link

	XI			**YJ**	(Individual)	
	XI:APn	U0:00		YJ:0Qo	V0:00	
	00:BQO	00:00		00:APN	00:00	
			* >>	YJ:0Qo	V0:E0	Assign1 (J)
				XI:APn	U0:00	C-on*
Assign1 (I)	XI:APn	US:CD	<< *			
C-on*, D-on	YJ:0Qo	V0:E0				
Go						
			* >>	YJ:BQo	VT:EF	D-on*
				XI:APn	US:CD	Go
	XI:APn	US:CD	<< *			
	YJ:0Qo	VT:EF				
			* >>	same		
	same		<< *			

Aggregated Link

	XI						YJ			
	XI:APn	U0:00	00:BQO	00:00		YJ:BQo	V0:00	00:APN	00:00	
					* >>	YJ:BQo	V0:00	XI:APn	U0:00	Assign (J)
										C-on>
						YJ:BQo	V0:E0	XI:APn	U0:00	<C!
Assign (I)	XI:APn	U0:00	YJ:BQo	V0:E0	<< *					
<C-on, D-on										
	XI:APn	U0:CD	YJ:BQo	V0:E0						
					* >>	YJ:BQo	V0:EF	XI:APn	U0:CD	D-on!
	XI:APn	U0:CD	YJ:BQo	V0:EF	<< *					
					* >>	same				
	same				<< *					
Go	XI:APn	US:CD	YJ:BQo	V0:EF						
					* >>	YJ:BQo	V0:EF	XI:APn	US:CD	
						YJ:BQo	VT:EF	XI:APn	US:CD	Go
	XI:APn	US:CD	YJ:BQo	VT:EF	<< *					