ANS Design Tools Cheat Sheet

There's a lot of stuff involved in Analogue Network Security. A few (awesome) reviewers told me to build an appendix; I said "How about a full-on Cheat Sheet?"

This cheat sheet aggregates key concepts, important formulas and charts, and reference data into one place. Please use your wits to help expand and refine ANS. This is just a start, and I look forward to your contributions. A printable version can be found at AnalogueNetworkSecurity.com.

THE BASICS

$$M1(v) > M2(v)$$

If Man 1 (from the Bear example), has greater velocity (a vector of speed and direction) than Man 2, Man 1 wins and Man 2 gets eaten by the bear. See OODA.

∞ is the enemy of security. If any of your designs allow ∞ in an answer, you have effectively zero security.

$$P(t) > D(t) + R(t)$$

For security to exist, protection time must be measurably and provably greater than the sum of Detection and Reaction Times. The goal is:

$$[D(t) + R(t)] \rightarrow O$$

$$D(t) + R(t) = E(t)$$

E(t) (Exposure Time) = the sum of Detection and Reaction Time. E(t) helps with calculating Trust Factors and Risk.

If $P(t) = 0$, then $D(t) + R(t) = E(t)$

If $P(t) < [D(t) + R(t)]$, then
$E(t) = \{[D(t) + R(t)] - P(t)\}$

DETECTION IN DEPTH

$P(t) > D(t) + R(t)$

^ ^

^ $P(rl) > D\ (rl) + R\ (rl)$

$P\ (dl) > D\ (dl) + R\ (dl)$

^

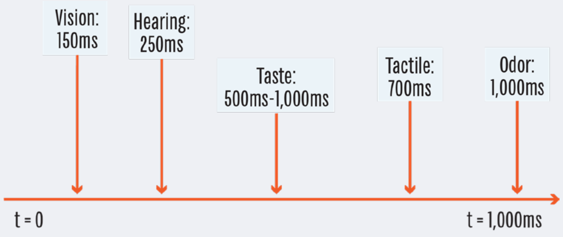
$P\ (d2) > D\ (d2) + R\ (d2)$

Adding security depth to Detection and Reaction channels.

Zeros We Love

D(t) → 0	E(t → 0
R(t) → 0	OODA(t) → 0

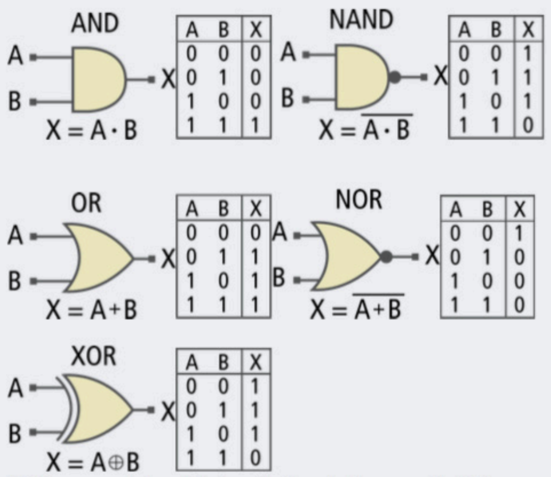
Range of Human Sense Dectection Times



Gives a sense of the range of human detection processes. How long does it take someone to "Click on Stupid Shit"? When we deal with humans, we need to calculate their time-values into equations.

BOOLEAN

Boolean logic and truth tables are essential to the hybridization of analogue and binary functions for ANS.



AND

A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

NAND

A	B	X
0	0	1
0	1	1
1	0	1
1	1	0

OR

A	B	X
0	0	0
0	1	1
1	0	1
1	1	1

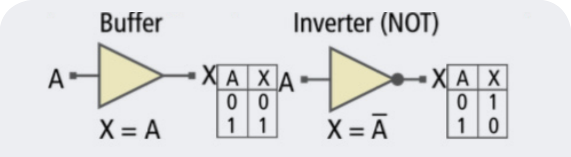
NOR

A	B	X
0	0	1
0	1	0
1	0	0
1	1	0

XOR

A	B	X
0	0	1
0	1	1
1	0	1
1	1	0

(XOR means only A or B, but not both, as is the case with OR.)



An inverter is just a buffer that flips the output value to the opposite of the input.

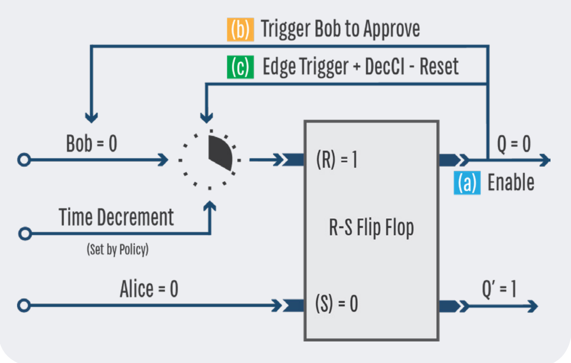
ANS BUILDING BLOCKS

When I designed electronic circuits, we had building blocks of components. It's the same thing with ANS. We rarely ever used hard math, either; it was 99% algebra. When we didn't know the design answer, we'd often w rite "T&C" next to a resistor on a schematic (T&C means "try and see" for yourself). We'd add a potentiometer, twist and tweak until we got the desired results, measure the values, and voila! We had the answer with no hard math.

When thinking analogue, getting close is often good enough (like horshoes or Bayes), and probably a far sight better than we are today. ANS designs employ lots of variables, some of which are policy based, measured processes, or based upon external or third party dynamic performance and behavior.

THE TIME-BASED FLIP-FLOP (TB-FF)

TB-FF Initial Default Conditions



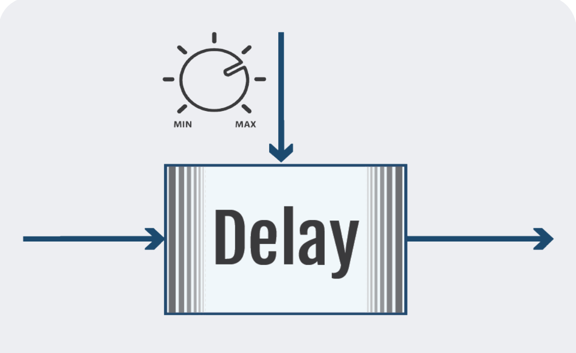
The Time-Based Flip-Flop is perhaps the most foundational aspect of ANS. Keep in mind that they can be concatenated, use independent or synchronous clocks, be combined in countless Boolean feedback networks.

This one circuit needs to be understood intuitively to maximize the power of ANS. The Truth Table *(top right)* may not be intuitively obvious until you use it a lot.

Truth Table: TB-FF

Alice (Set)	Bob (Approve)	Decrement (t)	Q = Enable
0	0	OFF	0
0	0	t > 0	0
0	0	t = 0	0
1	0	OFF	1
1	0	t > 0	1
1	0	t = 0	0
1	1	OFF	1
1	1	t > 0	1
1	1	t = 0	1
0	1	N/A	0
0	1	N/A	0
0	1	N/A	0

DELAY LINE



Time Delay: Adjust 0 ≤ Delay(t) <

The delay line variable is time, DL(t). In many processes, it's easy to show that DL(t) should simply be greater than E(t), which would then show P(t) > D(t) + R(t). Figure in physical layer latency and the human element as well into any process.

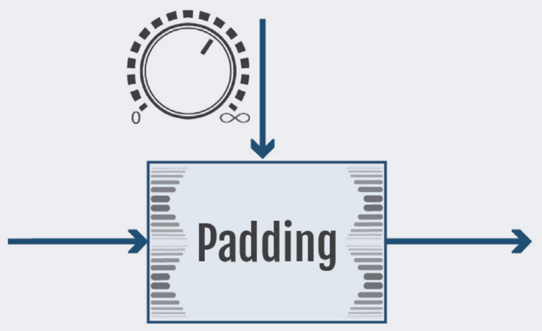
Pause/Play are conceptual triggers to be used in delay lines, especially with dynamic inputs. I imagine the time variable can be automatically adjusted with multiple weighting and potentially neural approaches.

DL(t): Delay Line (in time)

As a rule of thumb, if the introduced negative time > E(t) (justifiably > [D(t) + R(t)]),

security improves (and could be justifiable over time). Add Trust Factor for more complete answers.

PADDING



Data Padding: Adjust Pad where |DB| (Padded) = |DB| + |DB|* Pad%

Padding of data with "random" or "garbage" data for obfuscation increases the data-set size; thus it takes more time to be extricated.

Given same transmission bandwidth, the data exfiltration rates can be extended by Padding measured as:

100% < Padding Factor < ∞ %

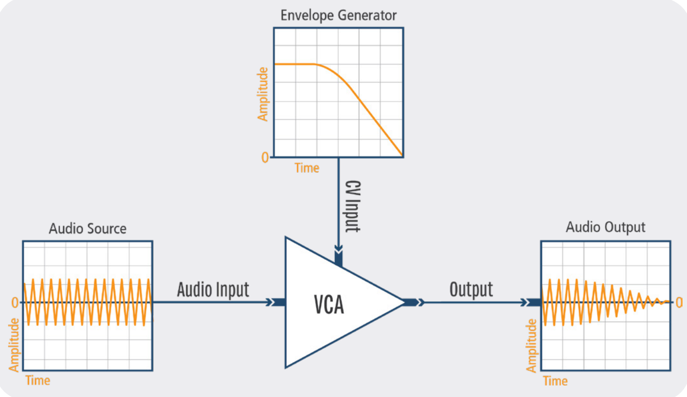
|DB|(2) = |DB|(1) * Padding (%)

Remember: IDBI/BW = max(E(t))

Bandwidth and data set size are time dependent.

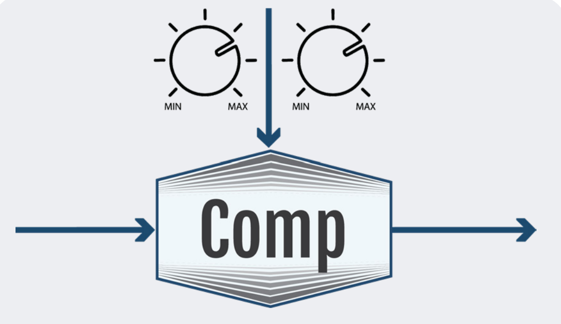
OOB CONTROL

The VCA approach is attractive for ANS Out of Band (OOB) controls. The audio circuit is analogous to TCP/IP data transmissions. The Envelop Generator and Control Voltage (CV) input is analogous to ANS style Out of Band security via a Detection/Reaction matrix.



Think of an OOB Analogue circuit as having its own C&C Server with detection in depth security controls embedded in the protocols to diminish the effects of attacks on the Detection and Reaction Matrix processes.

COMPRESSOR

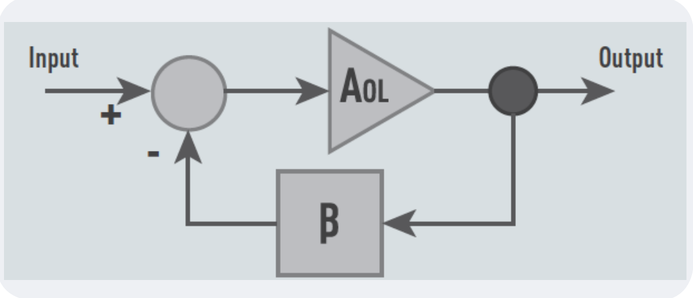


Compressor:
Adjust BW(min)>0 and BW(max)→∞

LIMITER

Bandwidth Compression and Limiting on specific data-rich services increases exfiltration time quickly. As part of a Reaction Matrix, the positive security effects can be exceedingly fast.

FEEDBACK



Without feedback, we approach infinity. We need limiting in the feedback loop. All control circuits should have feedback governors to maintain an upper-time-bound substantially less than ∞. Ideally, the feedback mechanism will be set to upper-bounds by policy, such as E(t-max), which defines the risk in time, and then we add Trust Factor.

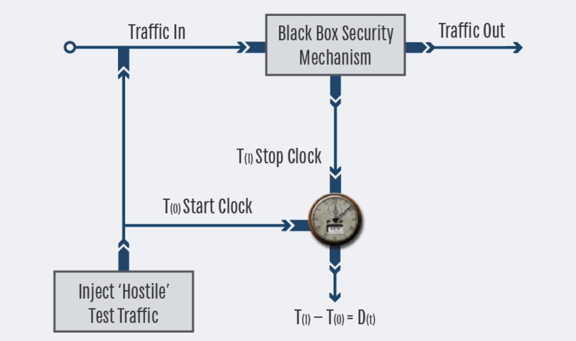
Negative feedback controls a system, while positive feedback creates runaway (growth) conditions. Oscillation between the two mechanisms is seen everywhere we look.

BLACK BOX

Security Black Boxes and controls "do" something, based upon one or more sets of input conditions. The output can be a shaped version of the input. A trigger output such as in detection applications, tells us when a Black Box event occurs. A gating function based upon control rules is also common.

From our views, we want to know the input(s), measure the output and output triggers in order to improve the efficacy of the process through feedback. The basic measurement is: **T(1) – T(0)** feedback networks.

Measuring Black Box Security

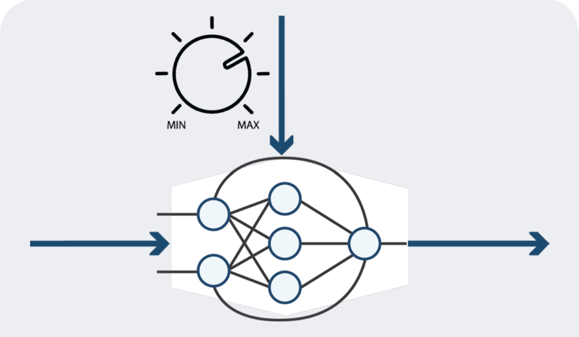

Traffic In
Black Box Security Mechanism
Traffic Out
$T_{(1)}$ Stop Clock
$T_{(0)}$ Start Clock
Inject 'Hostile' Test Traffic
$T_{(1)} - T_{(0)} = D_{(t)}$

From our views, we want to know the input(s), measure the output and output triggers in order to improve the efficacy of the process through feedback. The basic measurement is: T(1) – T(0) = D(t)

This one circuit needs to be understood intuitively to maximize the power of ANS. The Truth Table may not be intuitively obvious until you use it a lot.

NEURAL DECISION

In conceptual analogue design, consider replacing some fixed elements that require manual tuning, with some to-be-defined neural process. Small weighted networks that use dynamic information updates with variable Trust Factors assist with high speed decision making. Can be especially useful with time-based feedback processes. Adds variables, granularity and adjustable bias.



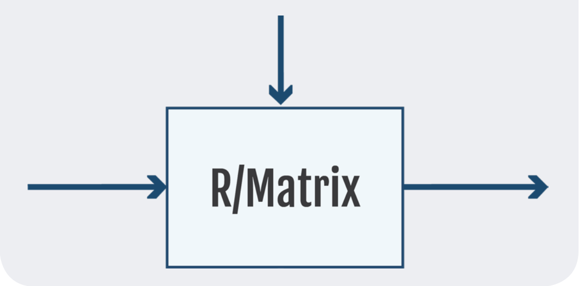
Neural Network: Adjust Bias 0 ≤ Bias ≤∞

MEMRISTOR

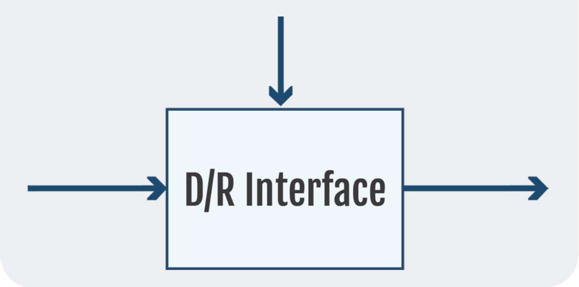
I suggest adding Memristor/Neural to your "Analogue News Feeds." I know this is wishful thinking, and a bit off in the future, but the potential is amazing.

ANS DETECTION & REACTION MATRICES

Reaction Matrix



Detection/Reaction Matrix



SET KNOB

Continuously variable control to set variables that come in two varieties (to keep it simple).

Min ≤ Set Knob ≤ Max

BAYES' THEOREM

Bayes is counter-intuitive, but we have to learn to live with that. The basic tenets are:

$$P(A \mid B) = \frac{P(B \mid A)\, P(A)}{P(A)}$$

where A and B are events and **P(B) ≠ 0.**

- ► **P(A)** and **P(B)** are the probabilities of observing A and B without regard to each other.
- ► **P(A) | P(B)**, a conditional probability, is the probability of observing event A given that B is true.
- ► **P(B) | P(A)** is the probability of observing event B given that A is true.

$$P(A \cup B) = P(A) + P(B) - P(A \cap B)$$

$$\begin{aligned} P(0.9 \text{ and } 0.9) &= P(0.9) + P(0.9) - P(0.9 * 0.9) \\ &= 1.8 - 0.81 \\ &= 0.99 \end{aligned}$$

In Boolean terms, this is an AND Gate, showing the increase in Trust Factor.

Example 1	Alice	Bob	Alice & Bob
TF	0.990	0.990	.9999
Risk	1.0000%	1.0000%	0.0100%
Risk Improvement			99.0000%

In this example with Alice & Bob, the increase in TF = 10^2.

$$
\begin{aligned}
P(A \cap B) &= P(TF(A) * TF(B)) \\
&= 0.9 * 0.9 \\
&= 0.81
\end{aligned}
$$

$$
\begin{aligned}
Risk &= 1-(P(TF(A) * TF(B)) = 1-(0.9*0.9) \\
&= 1- (0.81 \\
&= 0.91 = 1 - P(A \cap B)
\end{aligned}
$$

In Boolean terms, the OR gate reduces Trust Factor and increases Risk.

TRUST FACTOR

Trust is never absolute, so...

$$0 < \text{Trust Factor (TF)} < 1$$

As we get more granular, Trust Factors with six or more '9s' (0.9999999x) will be common. *(See the tables above.)*

Trust Factor with feedback will look like a sawtooth wave, bounded on the top with the unachievable "1" and on the bottom with a policy driven limit or a time-based reset/revet trigger.

SCALING

As we learn more about ANS, we will need to look at time-scaling for the future if any of this is going to be of long term benefit. We will be working with times from 10^{-12} to 10^{15} and beyond. Some of these charts will help you get a handle on the scales of ANS.

Computer Performance

Name	Unit	Value
kiloFLOPS	kFLOPS	10^3
megaFLOPS	MFLOPS	10^6
gigaFLOPS	GFLOPS	10^9
teraFLOPS	TFLOPS	10^{12}
petaFLOPS	PFLOPS	10^{15}
exaFLOPS	EFLOPS	10^{18}
zettaFLOPS	ZFLOPS	10^{21}
yottaFLOPS	YFLOPS	10^{24}

As discussed in the Fastest Computer, we will in the exa-flop/zetta flop range, sooner or later.

The laws of physics won't change, but our "cyber" will get much, much faster. One of the tenets of ANS is consider min-max at all times, because unbounded conditions yield an indeterminate and or infinity. Don't think slow. Prepare for fast. Faster. Faster than that.

(See Time & Clocks and Seconds tables on next page.)

# of Employees	# of Decisions	Per Time Period	Per 200 Work Days/Yr	Delta Six Sigma	Delta Six Sigma
				Trust Factor	Risk
10	10	Day	20,000	99.41%	0.59%
100	10	Day	200,000	94.12%	5.88%
1,000	10	Day	2,000,000	41.18%	58.82%
10,000	10	Day	20,000,000	-488.24%	588.24%
25,000	10	Day	50,000,000	-1370.59%	1470.59%
100,000	10	Day	200,000,000	-5782.35%	5882.35%
1,000	20	Day	4,000,000	-17.65%	177.65%
1,000	50	Day	10,000,000	-194.12%	294.12%
1,000	100	Day	20,000,000	-488.24%	588.24%
10,000	20	Day	40,000,000	-1076.47%	1176.47%
10,000	50	Day	100,000,000	-2841.18%	2941.18%
10,000	100	Day	200,000,000	-5782.35%	5882.35%
Six Sigma = 3.4*10^6	3,400,000				

6 Sigma vs. Trust Factor and Risk in different sized enterprises.

Time & Clocks

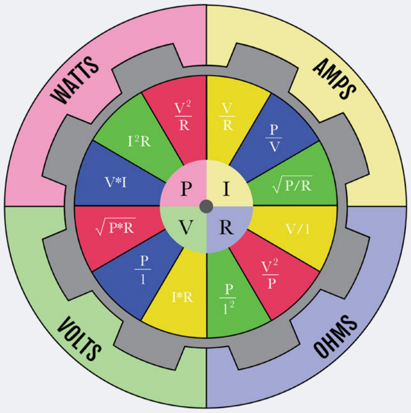
Notation	Seconds	Value	Time	Value	Time	Value	Time	Value	Time
10^0	1	0.02	Minutes						
10^1	10	0.17	Minutes						
10^2	100	1.67	Minutes						
10^3	1,000	16.67	Minutes						
10^4	10,000	2.78	Hours						
10^5	100,000	27.78	Hours	1.16	Days				
10^6	1,000,000	277.78	Hours	11.57	Days	1.65	Weeks	0.03	Years
10^7	10,000,000	2,777.78	Hours	115.74	Days	16.53	Weeks	0.32	Years
10^8	100,000,000	27,777.78	Hours	157.41	Days	165.34	Weeks	3.18	Years
10^9	1,000,000,000	277,777.78	Hours	11,574.07	Days	1,653.44	Weeks	31.80	Years

Seconds

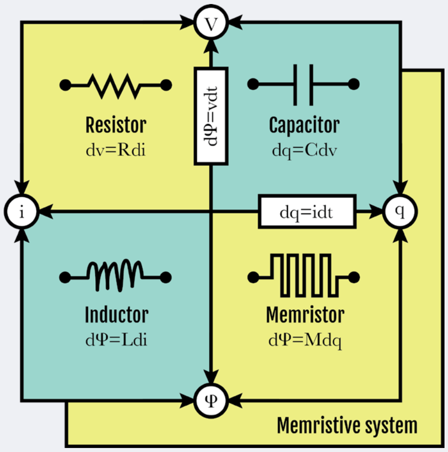
	Decimal	Scientific Notation
Day	86,400	8.64*10^4
Week	604,800	6.05*10^5
Month	2,592,000	2.59*10^6
Year	31,536,000	31.54*10^7

ELECTRICAL BASICS

Passive Electrical Component Quadrant



Calculating Arithmetic & Geometric Means



THE MEANS

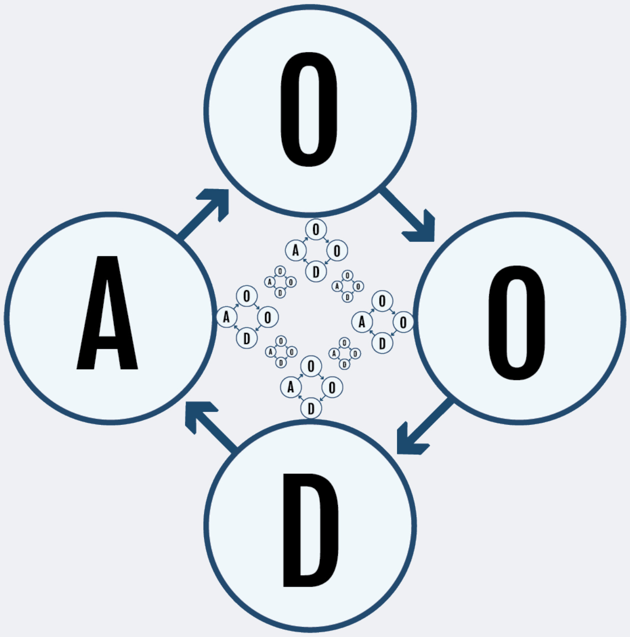
The standard average we learn at school is the Arithmetic mean. To find the average of 4 inputs we do the following calculation:

$$\frac{1}{n}\sum_{i=1}^{n} x_i \text{ and } \sqrt[n]{\prod_{i=1}^{n} x_i}$$

OODA LOOPS

OODA is core to design and operational security.

Sub-OODA (Granularity)



An OODA-loop can be three, four, five, or more iterative steps, each with it's own defined time and goals. Each step in the loop can and generally should have more granularized sub-loops (sub-processes) that should increase Trust Factor over time. Check out the formulas to the right.

$$L(t) = O1(t) + O2(t) + DE(t) + Act(t) = D(t) + R(t)$$

...where we want

$$L(t) \rightarrow 0$$

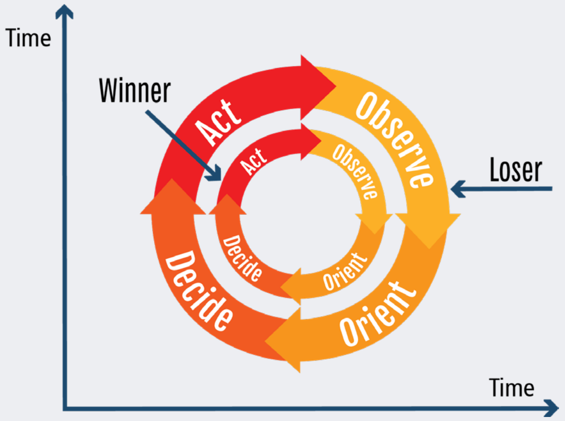
$$O1(t) = D(t)$$

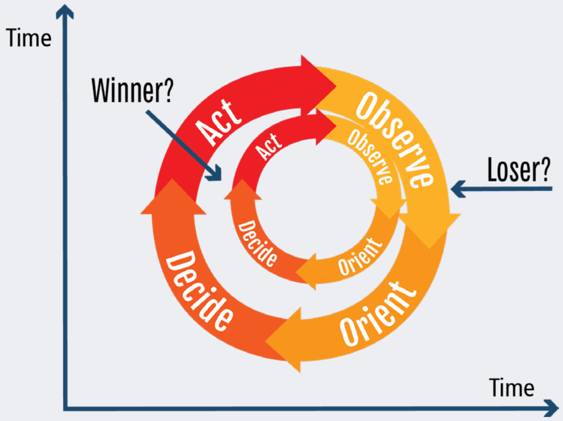
and

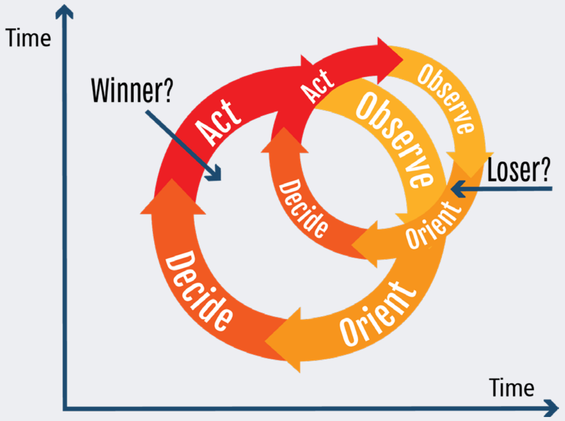
$$O2(t) + DE(t) + Act(t) = R(t)$$

DEFENSE: Go Faster. Measure More.

The following diagrams are effects of attacking and defending OODA loops in time.







If $A/L(t) > D(t) + R(t)$, Defense wins by $A/L(t) - [D(t) + R(t)]$, thus $L(t) < [D(t) + R(t)] < A/L(t)$

If $A/L(t) < D(t) + R(t)$, Offense wins by $[D(t) + R(t)] - A/L(t)$ thus $L(t) > [D(t) + R(t)] > A/L(t)$

If $A/L(t) < D(t) + R(t)$, then Offense wins as $E(t) = [D(t) + R(t)] - A/L(t)$