Response Time Goals: Average or Percentile

Scott Chapman

Enterprise Performance Strategies

Scott.Chapman@epstrategies.com

Contact, Copyright, and Trademarks

Questions?

Send email to performance.questions@EPStrategies.com, or visit our website at https://www.epstrategies.com or http://www.pivotor.com.

Copyright Notice:

Trademarks:

Enterprise Performance Strategies, Inc. presentation materials contain trademarks and registered trademarks of several companies.

The following are trademarks of Enterprise Performance Strategies, Inc.: **Health Check®, Reductions®, Pivotor®**

The following are trademarks of the International Business Machines Corporation in the United States and/or other countries: IBM®, z/OS®, zSeries®, WebSphere®, CICS®, DB2®, S390®, WebSphere Application Server®, and many others.

Other trademarks and registered trademarks may exist in this presentation

Abstract

WLM response time goals: we all love them if only because velocity goals are difficult to understand and maintain. But it turns out that response time goals have their own issues as well. In this presentation, we'll discuss response time goals and when you do and don't want to use them. We'll also compare average and percentile response time goals and when you might want to use each. Spoiler Alert: average response time goals can be useful in the modern mainframe environment!

EPS: We do z/OS performance…

- Pivotor - Reporting and analysis software and services
  - Not just reporting, but analysis-based reporting based on our expertise
- Education and instruction
  - We have taught our z/OS performance workshops all over the world
- Consulting
  - Performance war rooms: concentrated, highly productive group discussions and analysis
- Information
  - We present around the world and participate in online forums

Like what you see?

- Free z/OS Performance Educational webinars!
  - The titles for our Summer / Fall 2024 webinars are as follows:
    - ✓ *What a z/OS Guy Learned About AWS in 10 Years*
    - ✓ *Advantages of Multiple Period Service Classes*
    - ✓ *Understanding z/OS Connect Measurements*
    - ✓ *WLM and SMF 99.1 – System Measurements Deeper Dive*
    - ✓ *WLM and SMF 99.2 – Service Class Period Measurements Deeper Dive*
    - ✓ *Optimizing Performance at the Speed of Light: Why I/O Avoidance is Even More Important Today*
    - ✓ *Understanding MVS Busy % versus LPAR Busy % versus Physical Busy %*
    - ✓ *Rethinking IBM Software Cost Management Under Tailored Fit Pricing*
    - ✓ *Understanding Page Faults and Their Influence on Uncaptured Time*
    - *Response Time Goals: Average or Percentiles?*
    - *Understanding and Using Enclave*

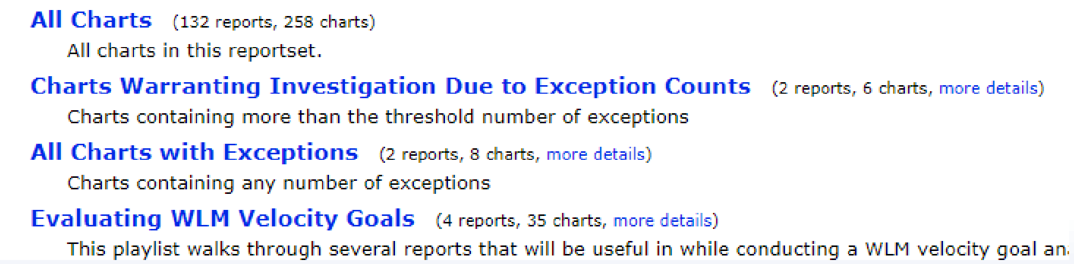
- If you want a free cursory review of your environment, let us know!
  - We're always happy to process a day's worth of data and show you the results
  - See also: http://pivotor.com/cursoryReview.html

Like what you see?

- The z/OS Performance Graphs you see here come from Pivotor™

- If you don’t see them in your performance reporting tool, or you just want a free cursory performance review of your environment, let us know!
  - We’re always happy to process a day’s worth of data and show you the results
  - See also: http://pivotor.com/cursoryReview.html

- We also have a free Pivotor offering available as well
  - 1 System, SMF 70-72 only, 7 Day retention
  - That still encompasses over 100 reports!



All Charts (132 reports, 258 charts)
All charts in this reportset.

Charts Warranting Investigation Due to Exception Counts (2 reports, 6 charts, more details)
Charts containing more than the threshold number of exceptions

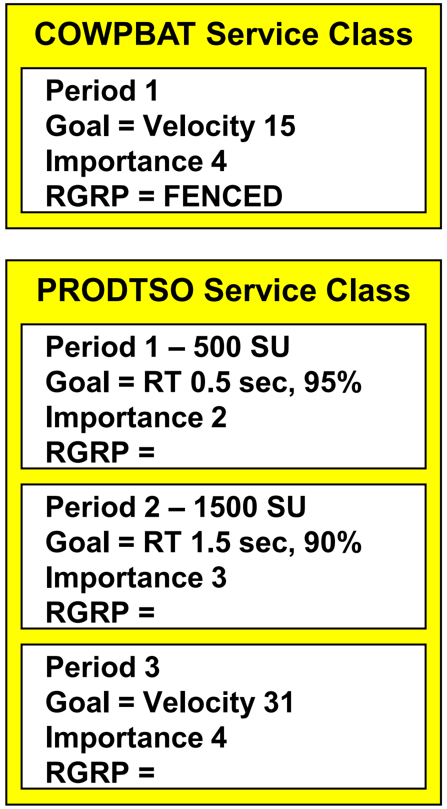
All Charts with Exceptions (2 reports, 8 charts, more details)
Charts containing any number of exceptions

Evaluating WLM Velocity Goals (4 reports, 35 charts, more details)
This playlist walks through several reports that will be useful in while conducting a WLM velocity goal an

WLM Terminology & Concepts

Service Classes (SC)

- Service Classes define work with similar:
  - Work types
  - Performance goals
  - Resource requirements
  - Business importance to the installation
- A service class consists of:
  - Service class name
  - Service class description
  - Period(s)
    - Performance goal and importance
    - Durations
  - Resource group name
- Service class can only be associated with one workload
- Can define up to 100 service classes



COWPBAT Service Class

Period 1
Goal = Velocity 15
Importance 4
RGRP = FENCED

PRODTSO Service Class

Period 1 – 500 SU
Goal = RT 0.5 sec, 95%
Importance 2
RGRP =

Period 2 – 1500 SU
Goal = RT 1.5 sec, 90%
Importance 3
RGRP =

Period 3
Goal = Velocity 31
Importance 4
RGRP =

Performance Index (PI)

- Every 10 seconds WLM calculates the PI for every service class period
  - PI is an indicator of how well a service class period is achieving its goal
  - Allows for comparison of unlike goals for unlike work
- PI < 1 indicates that a goal is being exceeded
  - example: PI = .5 means that work is achieving twice goal
- PI = 1 indicates that a goal is exactly being met
- PI > 1 indicates that a goal is being missed
  - example: PI = 3 means goal is being missed by 3 times

Performance Goals

- Performance goals are assigned to each period in a service class
  - All service classes have at least one period and each period has a goal
- There are four types of goals:
  - Response time goal
    - Indicates how quickly you want work to be processed
  - Velocity goal
    - Indicates the speed (or acceptable delay) for work
  - Discretionary goal
    - For low priority work for which you do not have any particular performance goal
  - WLM defined goals
    - Implied objectives of work WLM determines as needing special requirements
- WLM algorithms manages all work to one of these goal types

So What Type of Goal Should be Used?

- Response Time:
    - Wherever you can, assuming:
        - The goal is in the seconds range (or sub-second)
        - There are enough transactions completing: at least 10 in 20 minutes, preferably more
    - Gives you response time reporting for your transactions
    - Easy to understand

Special Note: RT goals are great for DDF, *unless* you're using hi-perf DBATs. See: "WLM Management of DDF Work: What can you do and what has changed?"

- Discretionary:
    - Work that can wait until other work is done
    - Last "penalty" period for certain workloads
    - Note that if the system is constrained, discretionary work will be mostly delayed
- Velocity:
    - Use for everything else (most STCs, most batch) that shouldn't be in SYSSTC or SYSTEM

What is a Transaction

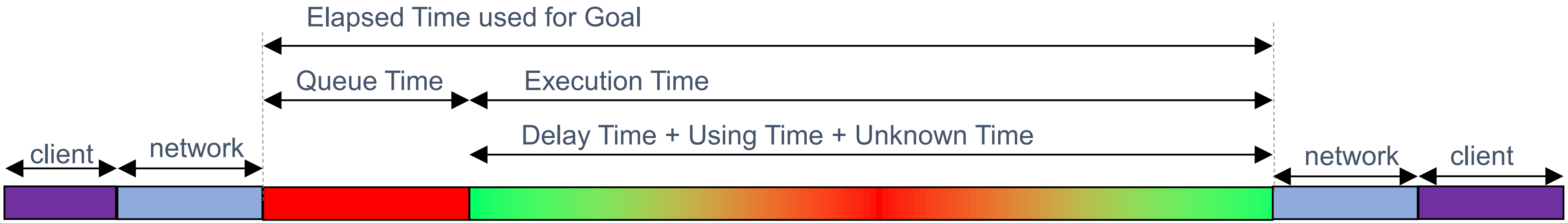
- Transaction
  - A way of delineating a unit of work that is consuming service
- WLM associates performance characteristics to said unit of work
  - Thus, it is the transaction that has the performance characteristics and requirements…
  - …which is not necessarily the same as the address space(s) processing the transaction
- Examples of transactions
  - CICS or IMS transactions
  - TSO
    - Usually corresponds to a command or terminal interaction
  - DDF
    - Start of connection (or prior commit) to commit/abort, can be 1 or many SQL statements
  - Batch transactions
    - Corresponds to a job execution
  - IBM Apache Web Server (web server)
    - A web request (i.e. request to server a html file or jpeg file, or run a cgi or plug-in
  - Started Task
    - Generally, the life of the address space

Response Time Goals

Averages and Percentiles

Response Time Goal Components

- Transaction response time includes
  - Managed Queue time
    - Wait for a WLM-managed JES initiator
    - Wait for an APPC initiator
    - WebSphere Application Server – Waiting for a thread in a servant region (AE queue)
    - Wait for logon, or logon proceeding
  - Execution time
    - Known using time
    - Known delay time
    - Unknown time
- A fairly accurate reflection as to what was achieved on z/OS (vs end to end RT)


Elapsed Time used for Goal
Queue Time
Execution Time
Delay Time + Using Time + Unknown Time
client
network
network
client

Average Response Time Goals

- The average response time desired for a given set of ended transactions
- Response time as measured by WLM

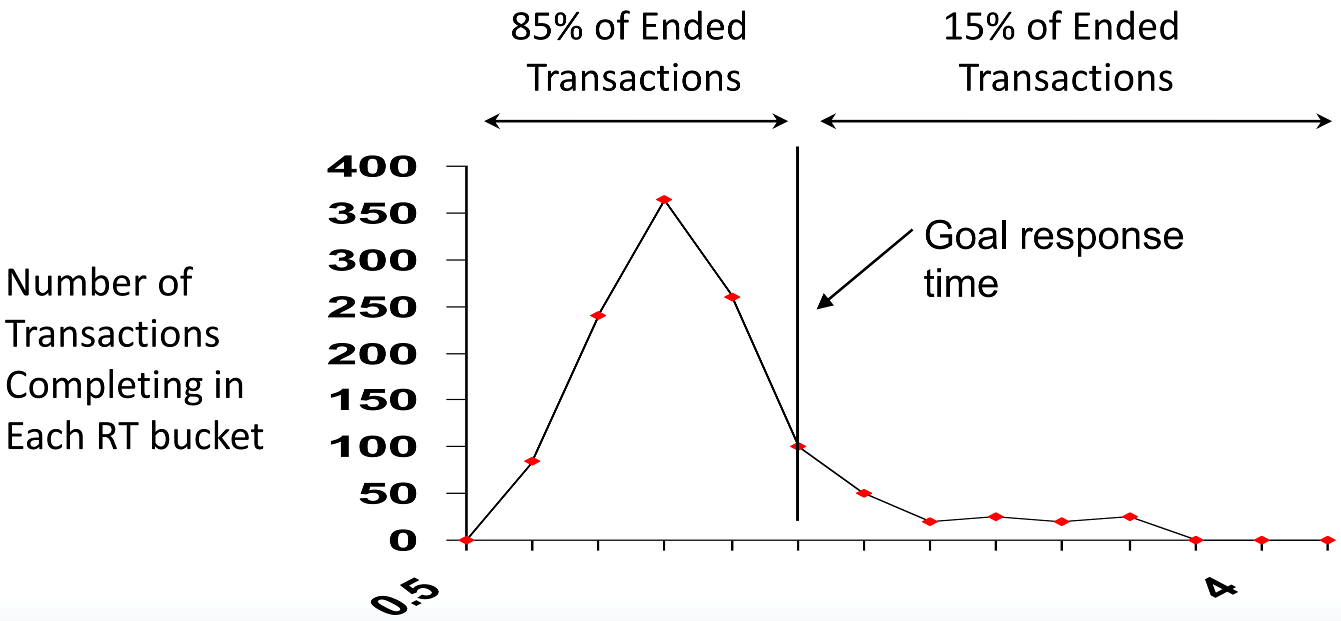
$$avg\ response\ time = \frac{sum\ of\ elapsed\ times}{number\ of\ ended\ transaction}$$

- Average response time goals can be easily influenced by 'outliers’:
  - Average response time goal set to 1 second
  - 99 transactions complete in 1 second
  - 1 transaction completes in 2 minutes
  - Average response time achieved is 2.2 seconds
  - Goal missed even though 99% of transactions completed within 1 second

Percentile Response Time Goals

- Percentile of ended transactions that need to complete within a particular response time desired
  - Reduces the influence of outlier transactions
    - Example: 85% of transactions (or better) to complete within a given response time means we don't really care about the longest 15% of transactions
  - WLM can manage to the "typical" transaction

15% seems like a significant portion to ignore which is why percentile goals >= 90% are generally preferred!



Response Time PI Formulas

$$\text{Average RT Goal PI} = \frac{\text{Actual Average Response Time}}{\text{Average Response Time Goal}}$$

Example

- Actual response time average of 0.1 seconds
- Average response time goal of 0.5 seconds
- PI = 0.1 / 0.5 = 0.2 = greatly over-achieving its goal

$$\text{Percentile RT Goal PI} = \frac{\text{Actual RT at Percentile}}{\text{Response Time Goal at Percentile}}$$

Understanding WLM RT Distribution

- WLM maintains a response time distribution for periods assigned a response time goal (both types)
  - Distribution compose of 14 buckets
  - Each bucket represents a count of transactions that completed within a certain percentage of the assigned goal value
    - Bucket 4 represents count of transactions completing between 70% and 80% of the goal value
    - Bucket 6 represents count of transactions completing between 90% and exactly the goal value
    - Bucket 12 represents count of transactions that complete between 1.5 and twice the goal value
    - Bucket 13 represents count of transactions that complete between twice and 4 times goal value

Bucket	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Width	<=50%	60%	70%	80%	90%	100%	110%	120%	130%	140%	150%	200%	400%	>400%
Transaction Count	0	85	240	365	260	100	50	20	25	20	25	0	0	0

More RT Distribution Details

- The RT value of each bucket is dependent on the goal
  - The below example is a distribution for a 2 second response time goal
- Buckets 1 and 14 are unique in that they are unbounded
  - We never know the precise time range that the transactions completed in
  - I.E. bucket 14 could contain transactions completed in 5x, 10x, or 100x the goal value
- Response time distribution data is reported by the performance monitors

Bucket	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Width	<=50%	60%	70%	80%	90%	100%	110%	120%	130%	140%	150%	200%	400%	>400%
Value	<=1sec	1.2sec	1.4sec	1.6sec	1.8sec	2sec	2.2sec	2.4sec	2.6sec	2.8sec	3sec	4sec	8sec	>8sec
Trans Count	0	85	240	365	260	100	50	20	25	20	25	0	0	0

Percentile RT PI Details

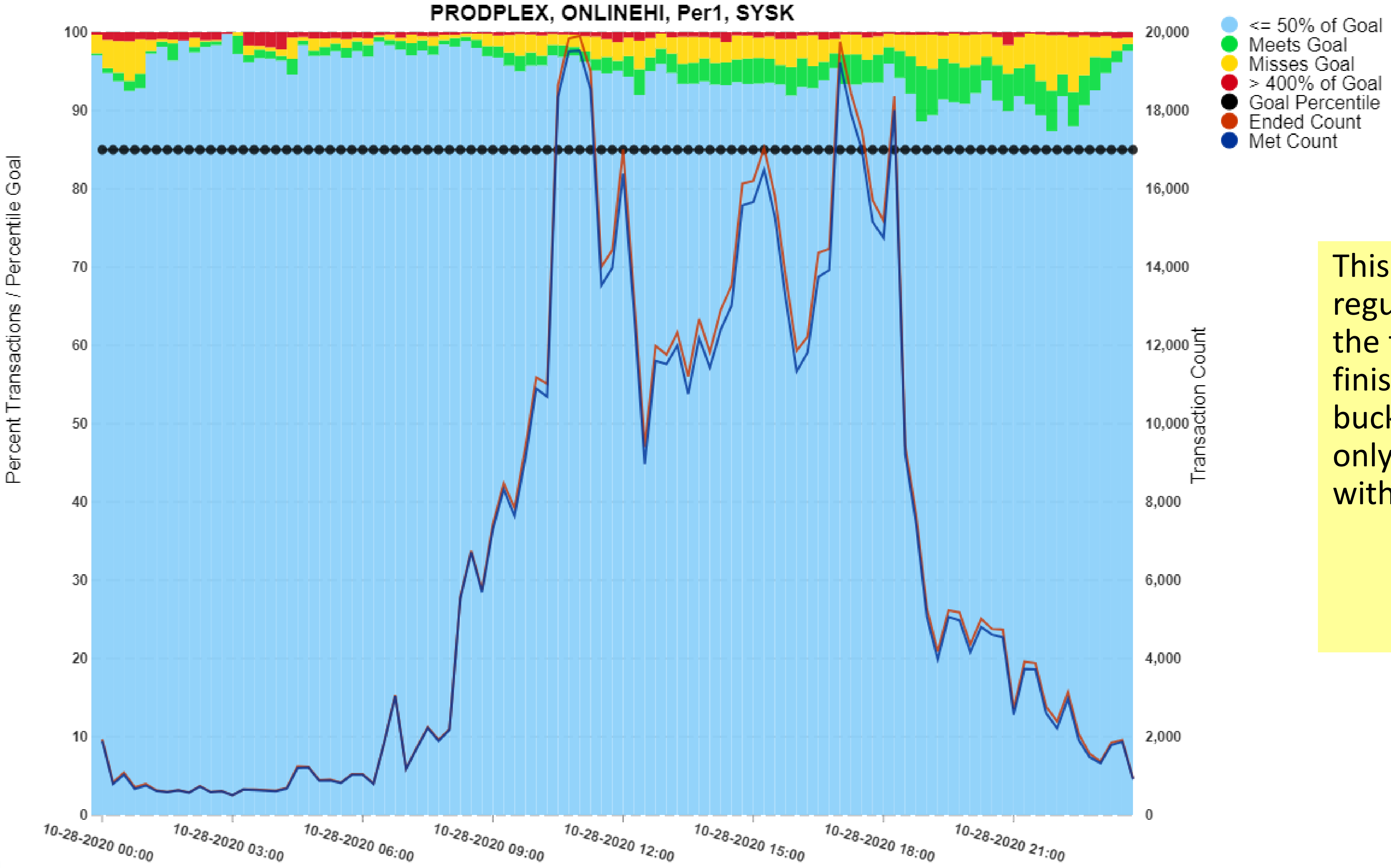
- To calculate the PI for a percentile RT goal we need response time at percentile
  1. Determine total number of completed transactions (add all buckets)
  2. Using the percentile objective, calculate the number of transactions that equal that percentage
  3. Add buckets 1 to n until you get a transaction count of at least that calculated in step 2
  4. Calculate PI by dividing the response time represented by the nth bucket by goal response time
     - Note PI = bucket width

Bucket	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Width	<=50%	60%	70%	80%	90%	100%	110%	120%	130%	140%	150%	200%	400%	>400%
Value	<=1sec	1.2sec	1.4sec	1.6sec	1.8sec	2sec	2.2sec	2.4sec	2.6sec	2.8sec	3sec	4sec	8sec	>8sec
Trans Count	100	85	240	365	260	100	50	20	25	20	25	0	5	5
Cumm Count	100	185	425	790	1050	1150	1200	1220	1245	1265	1290	1290	1295	1300

- Example: Goal = 90% within 2.0 seconds
  1. Total completed transaction (in above distribution) = 1300
  2. 90% of 1300 = 1170
  3. When add buckets 1 to n we find the 7th bucket brings us to 1200 (just past goal value)
  4. The 7th bucket represents 110% of goal of 2 seconds or a PI of 1.1

     2.0 * 1.1 / 2.0 = 2.2 / 2 = 1.1

WLM RT Goal - RTD% of Trans Met/Missed RT Goal with Number Trans

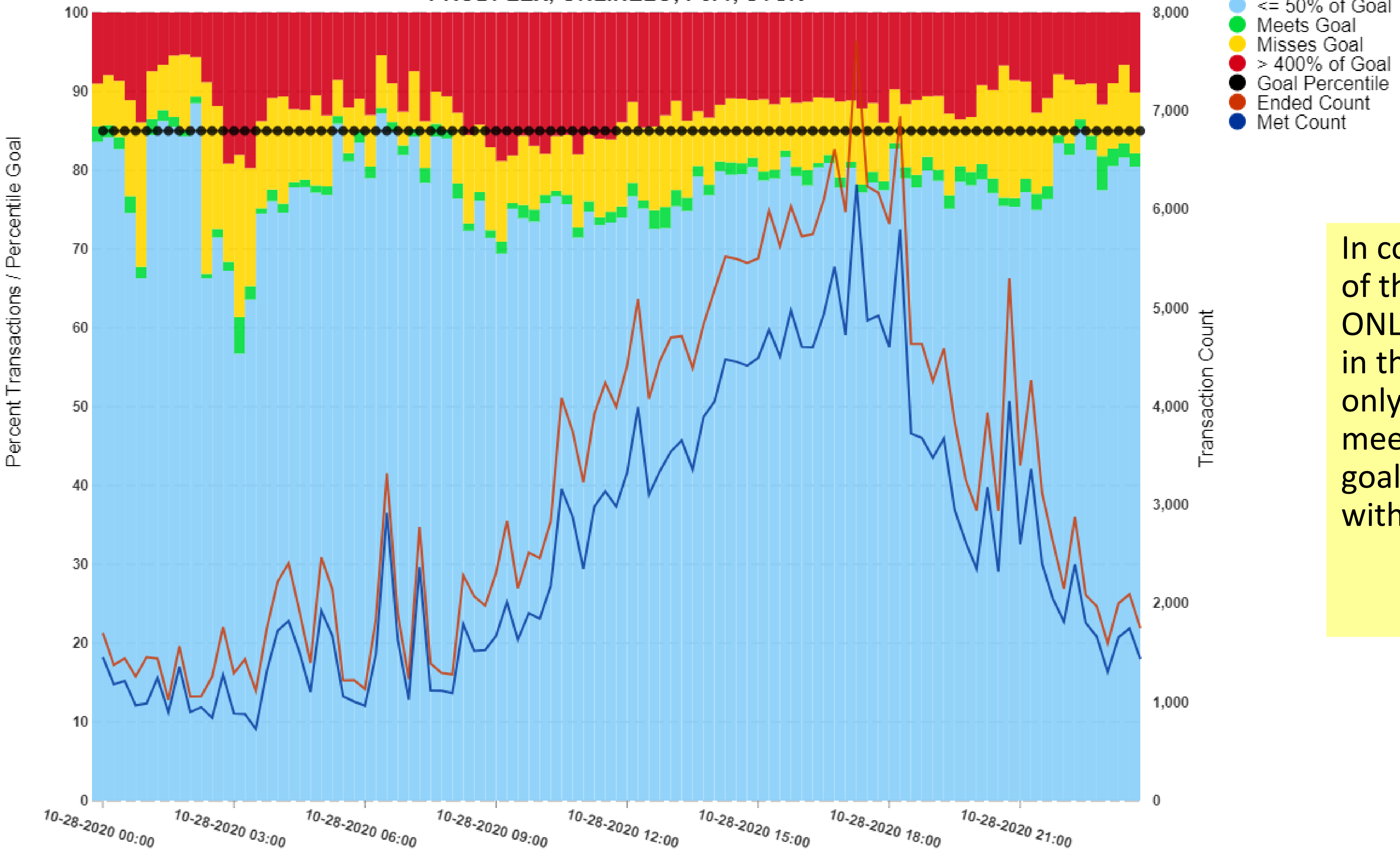


This Percentile RT Goal regularly has 90-95% of the transactions finishing in the first bucket, but its goal is only 85% complete within 0.25 seconds.

WLM RT Goal - RTD% of Trans Met/Missed RT Goal with Number Trans

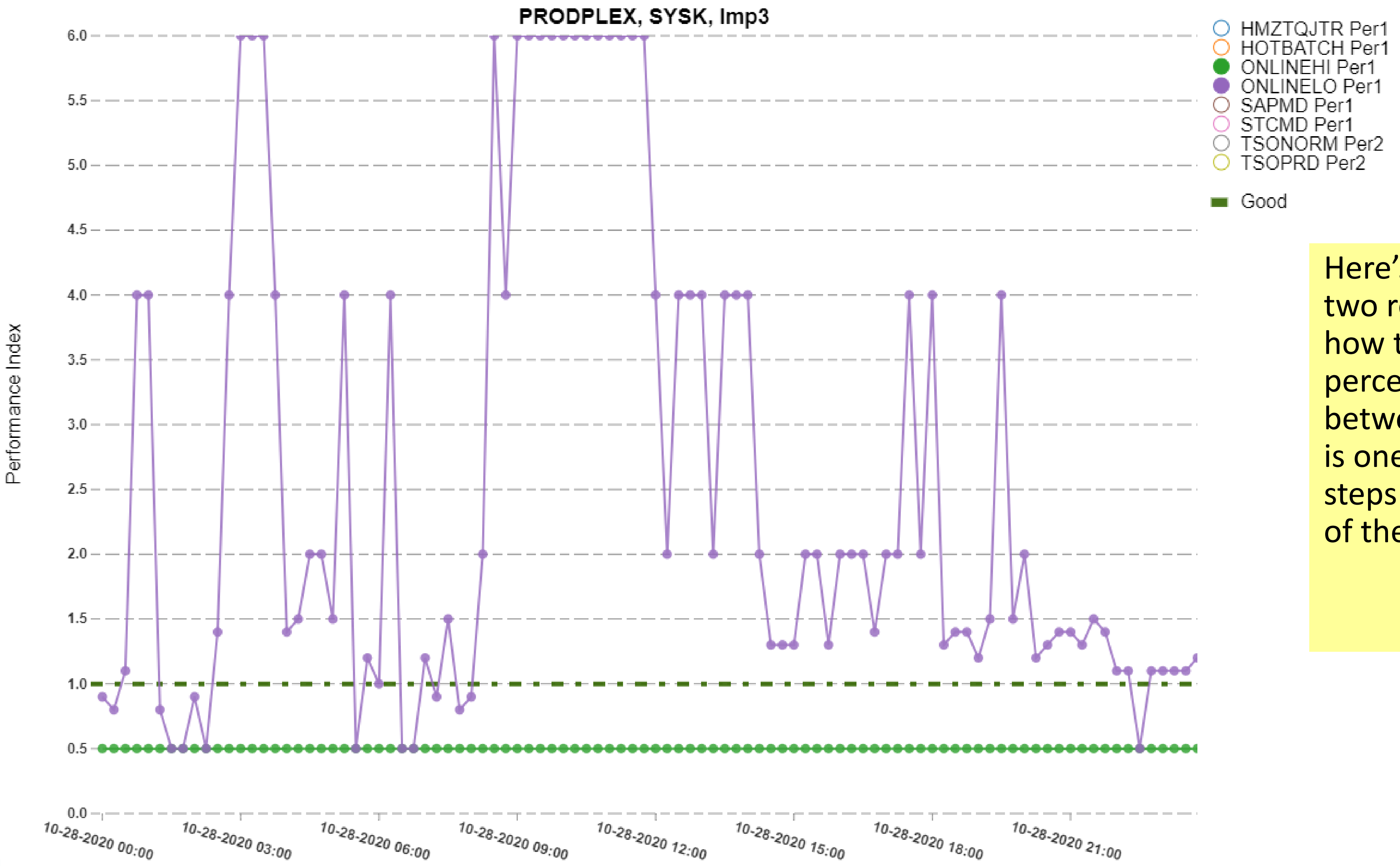
Percent met/missed goal and count

PRODPLEX, ONLINELO, Per1, SYSK



In contrast, here 10-15% of the transactions for ONLINELO are finishing in the last bucket, and only about 75-80% are meeting the goal, vs. the goal of 85% completing within 1 second.

WLM PI - For Importance for System (capped at PI=6)



Here’s the PI for those two report classes. Note how the PI for a percentile goal is always between 0.5 and 6. And is one of those distinct steps representing one of the 14 buckets.

Elapsed Time Trends

What was different 30 years ago?

- The mainframe has changed dramatically in the last 30 years!
  - First CMOS machine: 9672-R11: 696 SU/sec
  - Last bipolar machine: 9021-711: 3,018 SU/sec
  - *Smallest* z15 T02: 8562-A01: 5,022 SU/sec
  - Full speed z15: 8561-701: 103,488 SU/sec

Single Engine SU Ratings

- A few GBs of memory was a very large machine in the early 90s
  - Minimum z16 A02 memory is 64GB (z16 A01 minimum = 512GB!)
- IBM RAMAC Array DASD introduced in 1994
  - IBM ESS "Shark" was introduced in 1999
- SSD was not in widespread use

total DASD shipments are expected to increase 23% to 900TB this year and then rise another 33% to 1200TB in 1995

ComputerWorld Dec94

Alamo paid 'in the $3-and-change range [per MB]'

ComputerWorld Dec94

Both CPU and I/O are much, much faster than 20-25 years ago
Recommendations always need to be revisited as technology changes

How have 30 years affected RT Goals?

- New types of transactional work
  - DDF was much less prevalent to non-existent 20+ years ago
  - Websphere Application Server wasn’t a thing
- Elapsed time of transactions can be much faster
  - Much faster CPUs, much faster I/O, larger memory to avoid I/O
  - But data is also larger counter-acting some of that
- Elapsed time in some cases may be more consistent (higher n-way LPARs)
  - Of course in some cases, transactions have become less consistent (e.g. ad hoc DDF work from end users)
- Applications are more complicated
  - Relatively few transactions are pure 3270 transactions
  - Larger payloads (XML) from some transactions
  - Client-side response time sometimes significant
  - Multiple MF transactions combined to single end-user interaction
  - Potentially larger difference between transaction ET and end-user response

Fast transactions

- Transactions down to single-digit milliseconds are becoming common
  - Not everything, but certainly some things!
  - RT goals < 15 ms possible with z/OS 2.3 – take advantage of this when needed!

- Historically, we've not used RT goals for batch because batch jobs are generally not short-running transactions
  - But now: some customers have many batch jobs with elapsed time < 10 seconds
  - It may or may not make sense to have a RT goal for these sort of jobs
    - Likely a larger variety of elapsed times for a given CPU time due to I/O
  - If you're going to use RT for batch goals, make sure you have enough ending jobs!!

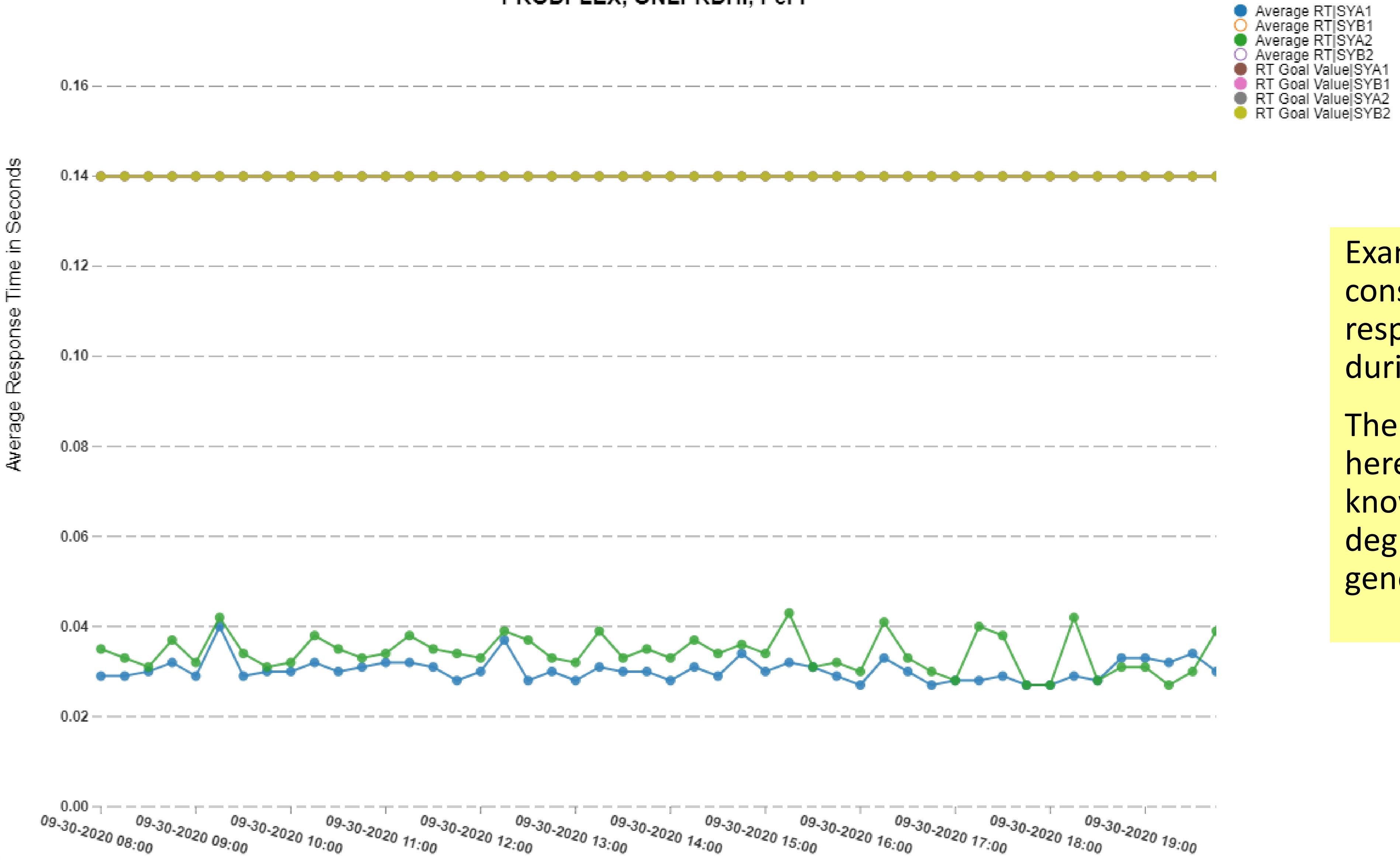
Elapsed Time Consistency

- Elapsed times can potentially be more consistent today
  - More/faster CPs = less CPU delay = less variation
  - Eliminated I/O queues = less variation in I/O times
- SCPs with homogenous transactions may have very consistent RTs
  - Homogenous = doing roughly the same work
  - Especially for high-importance/priority work
- “Well-behaved” applications may also have very consistent average RTs
  - Maybe the work isn’t homogenous, but the mix of work is consistent
- QMF is DDF work that’s likely to be neither homogenous nor consistent

WLM RT Goal - Average Response Time by Period

(Y-axis limited to 4 seconds)

PRODPLEX, ONLPRDHI, Per1



Example of a pretty consistent average response time (at least during the daytime).

The goal seems high here, but I happen to know that in this case degrading to 140ms was generally acceptable.

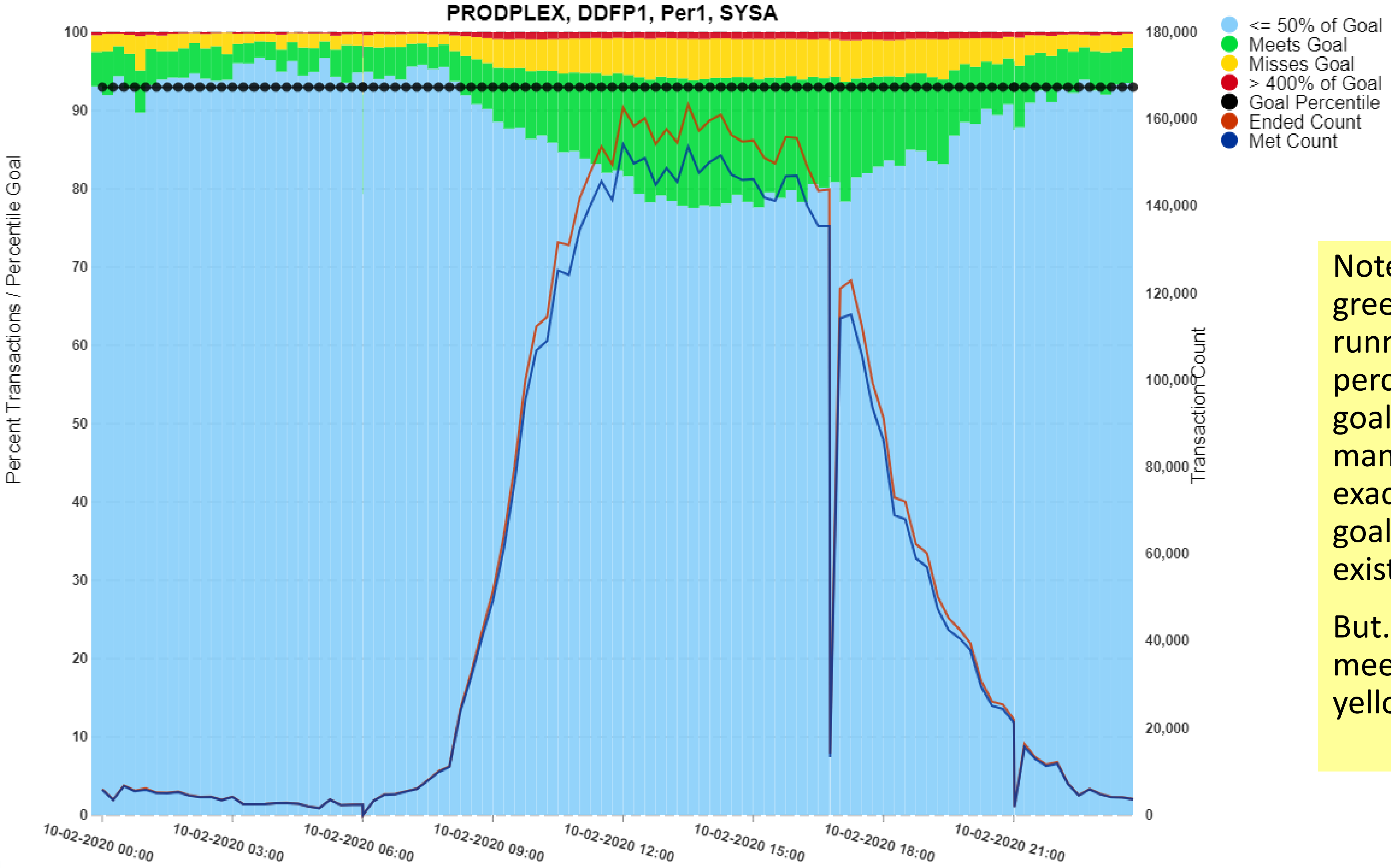
Percentiles or Averages?

Percentiles vs. Averages

- Percentiles often recommended to avoid impact of outliers
  - Do you have outliers that you need to ignore?
  - Do you want to ignore the outliers?
- Do you have lots of transactions (hundreds/sec or more)?
  - Are outliers really a problem in that scenario?
    - A few outliers are likely to be mitigated by the thousands of non-outliers
  - 1% of transactions at such rates can be a whole lot of transactions
- Do you have strata of transactions?
  - Many very short transactions
  - Significant longer transactions
    - Percentile goals may effectively ignore those longer running transactions
- Averages can let your goal be more sensitive to performance changes

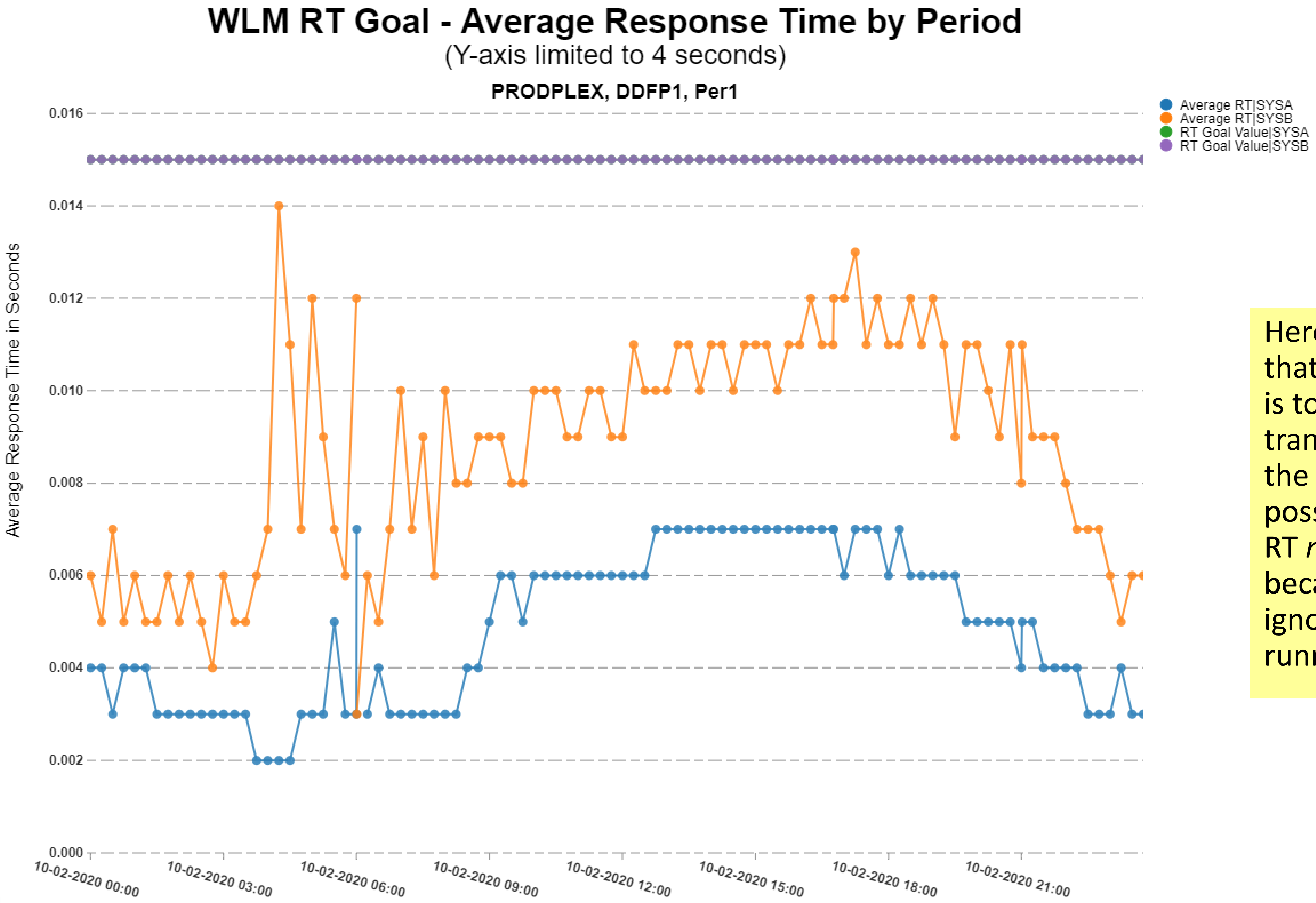
WLM RT Goal - RTD% of Trans Met/Missed RT Goal with Number Trans

Percent met/missed goal and count



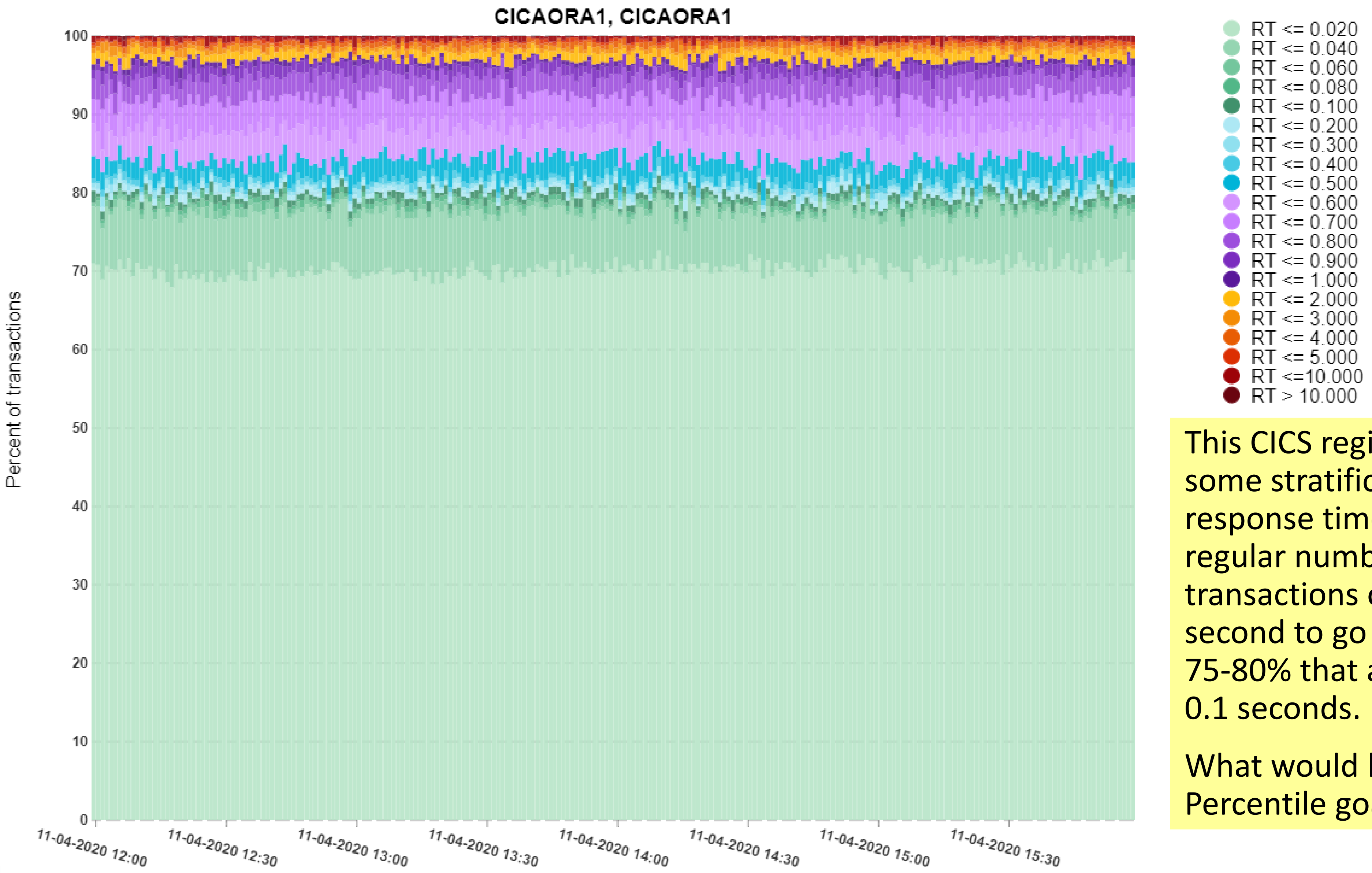
Note the top of the green ("meets goal") is running right at the goal percentage for this RT goal. Either WLM is managing this work exactly to goal or the goal was set to match existing performance.

But… this would still be meeting goal if all the yellow turned red.



Here's the average RT for that SCP. If our objective is to protect the transactions that were in the yellow band, it's possible that an average RT *might* be better because it wouldn't ignore those longer-running transactions.

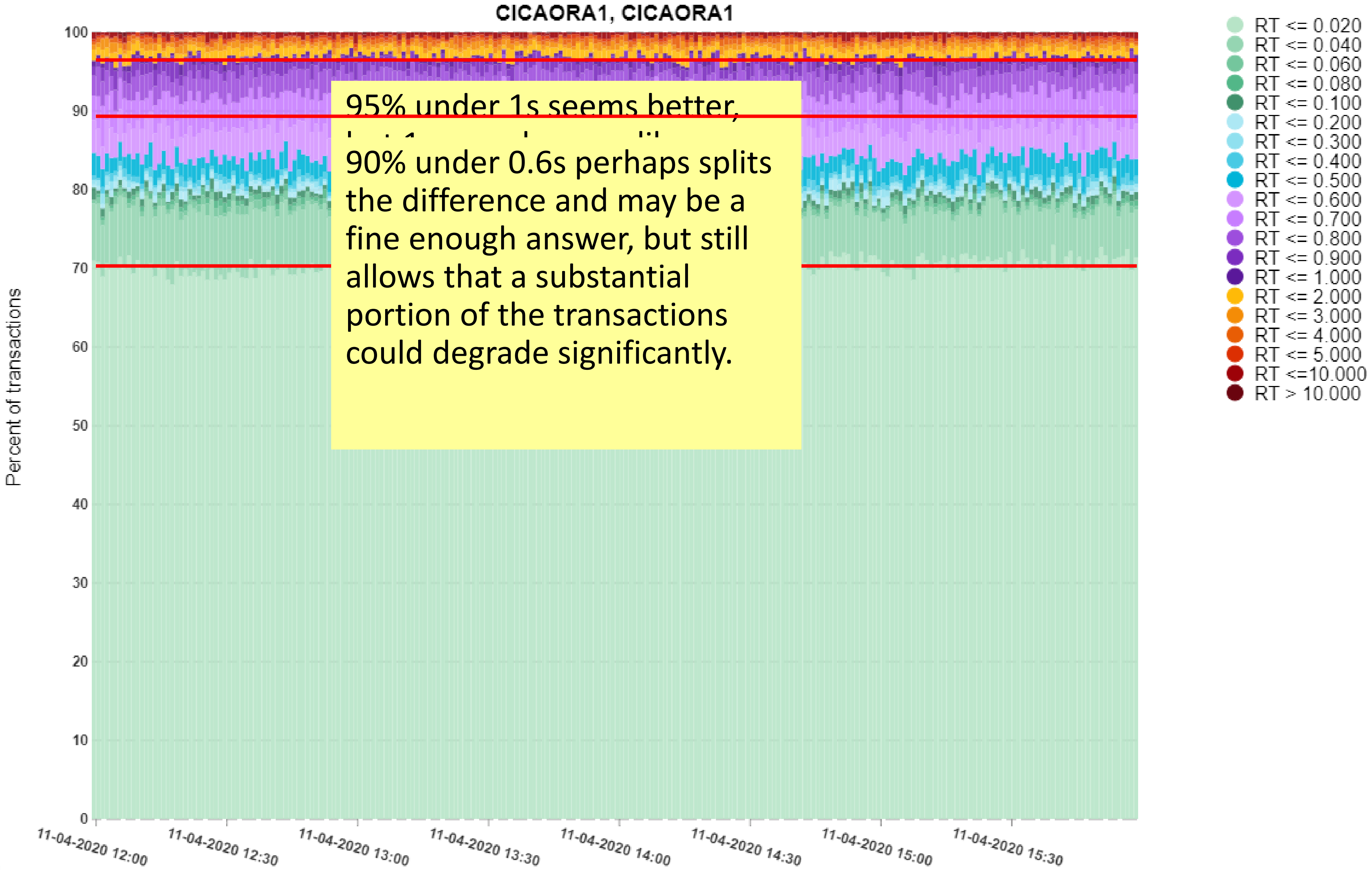
CICS Region Response Time Distribution



This CICS region shows some stratification of response times; there's a regular number of transactions over 1 second to go with the 75-80% that are under 0.1 seconds.

What would be a good Percentile goal here?

CICS Region Response Time Distribution



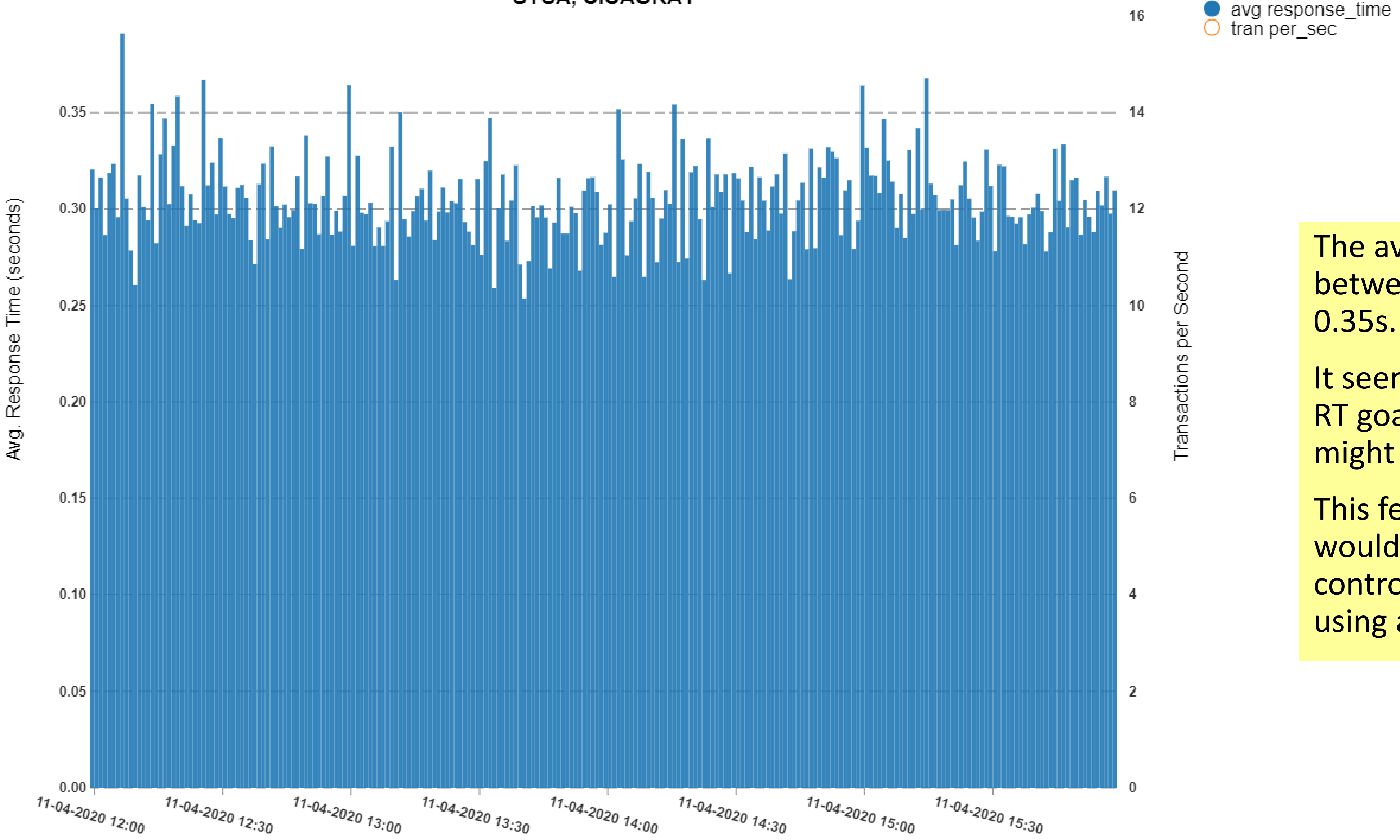
95% under 1s seems better, [illegible]

90% under 0.6s perhaps splits the difference and may be a fine enough answer, but still allows that a substantial portion of the transactions could degrade significantly.

Avg Response Time and Rate

(Excluding system transactions)

SYSA, CICAORA1

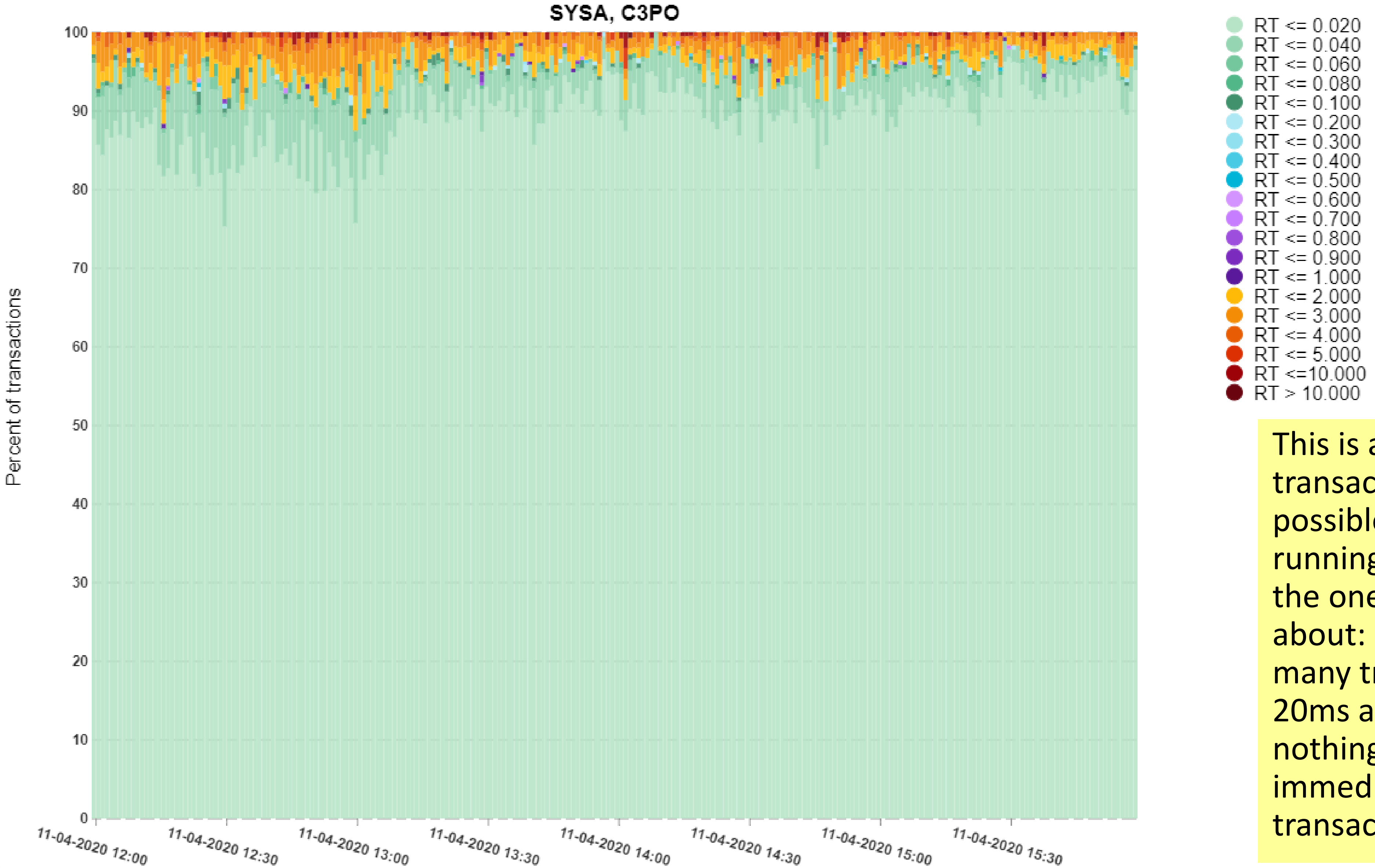


The average RT runs between about 0.25 and 0.35s.

It seems like an average RT goal of around 0.3s might be good.

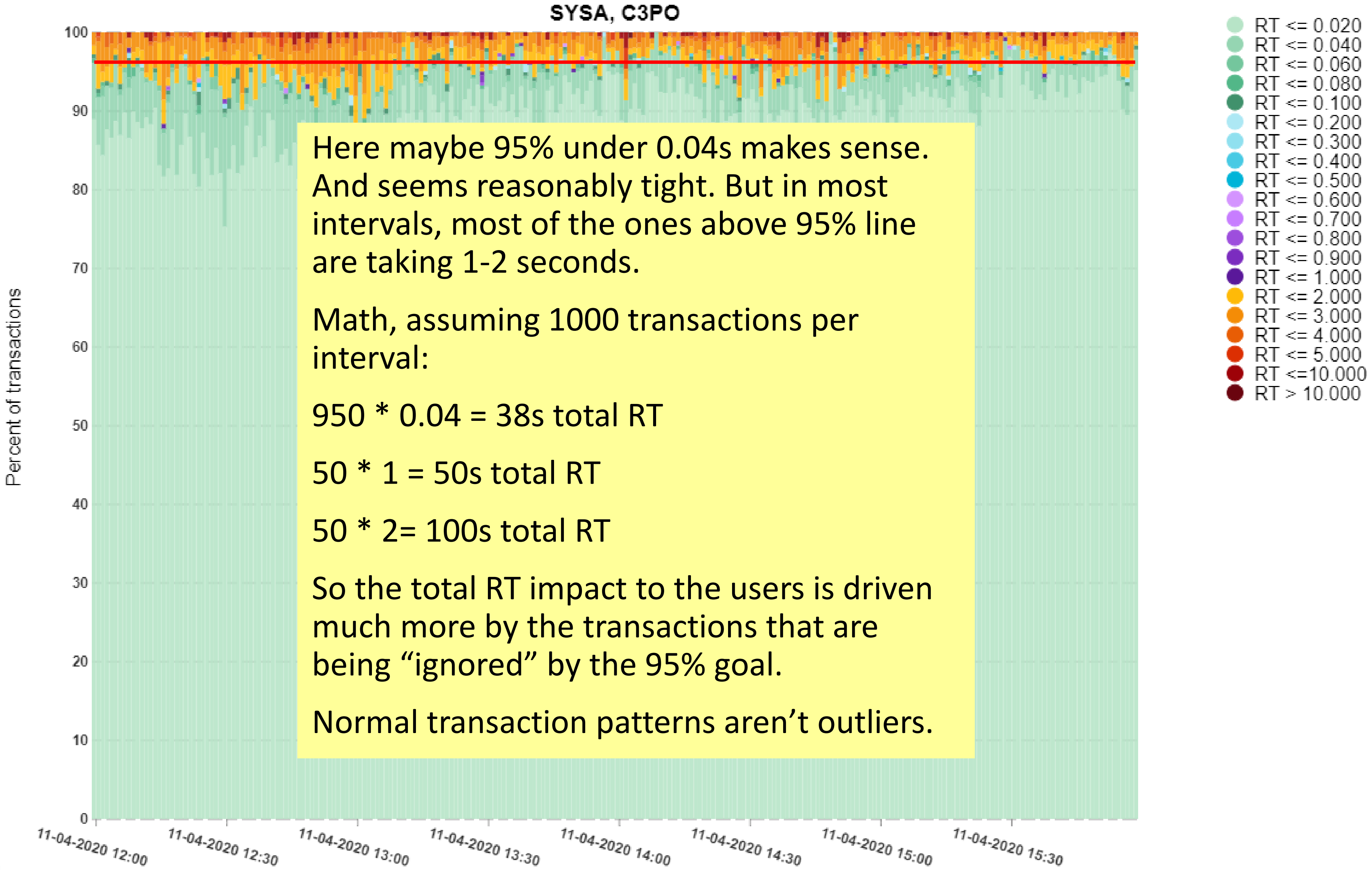
This feels to me like it would more closely control the work vs. using a percentile goal.

CICS Trans Response Time Distribution



This is a for a specific transaction id. It's quite possible that the longer running transactions are the ones we care more about: it's possible those many transactions under 20ms are really doing nothing but a return immediate to another transaction.

CICS Trans Response Time Distribution



Here maybe 95% under 0.04s makes sense. And seems reasonably tight. But in most intervals, most of the ones above 95% line are taking 1-2 seconds.

Math, assuming 1000 transactions per interval:

950 * 0.04 = 38s total RT

50 * 1 = 50s total RT

50 * 2= 100s total RT

So the total RT impact to the users is driven much more by the transactions that are being “ignored” by the 95% goal.

Normal transaction patterns aren’t outliers.

More percentile vs. averages thoughts

- Remember: the reason we set goals is to help WLM manage the work
- A more sensitive goal doesn’t help if the slowdown is beyond WLM’s control
- A slowdown due to the work having the wrong dispatching priority will likely impact both the short and long-running work
  - So a percentile goal set close to normal distribution should hopefully capture that
  - But longer transactions may be impacted more than shorter ones
  - And shorter ones could degrade (on average) substantially but not impact the percentiles if the longing running ones are long because of other reasons
- Average goals that occasionally spike due to outliers may cause WLM to chase problems it can’t (and doesn’t need to) help

Setting RT Goals Summary

- Generally avoid “loose” percentile goals—especially for large volumes
- The outlier transaction problem may not be the same as it used to be
  - Some transactions will necessarily take longer (they may also be more important)
- Average RT Goals will react to increased response time across the entire population of transactions
  - Percentiles can ignore changes above/below the goal (which may be ok, or not)
  - While not appropriate for all situations, averages are worth considering
- Understand your applications and understand what your users wait for!
  - Are they waiting on one transaction or multiple?
  - Is the network/client time significant compared to the mainframe ET?
  - “As fast as possible” may not be the most financially justifiable
  - Are your users even users or a batch process?

Thank you!

If you have any questions, feel free to ask them now

Or email me later at scott.chapman@epstrategies.com