ASAP 2.4
ASAP Extension 2.4

Monitoring Super-Scalar Systems

joe.davis@hp.com
http://NonStopASAP.com

ASAP 2.4

Can you monitor? 10,000 processes in a processor?

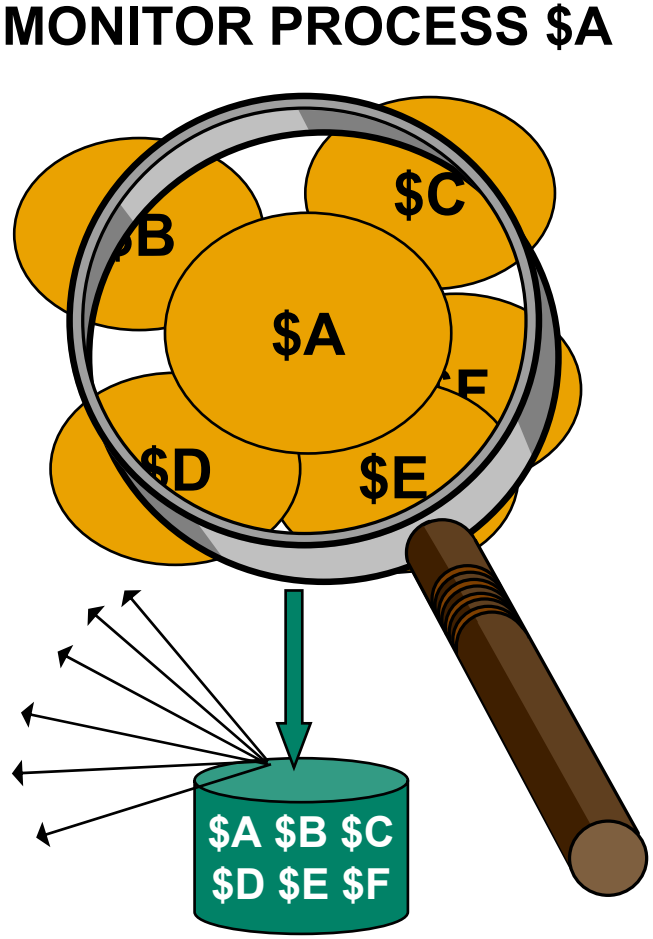
160,000 processes in a system?

40,800,000 processes in an Expand network?

- That's what's coming! Super-fast, super-scalar systems
- Requirements:
  - Must analyze each individual process and compare against predefined service levels, alerting when necessary
  - Must be able to group processes and view at the group level
  - Must be able to aggregate/propagate data and alerts to the group level
  - Must be able to set service level objectives at aggregate levels
  - Must be able to store summary information about the group for historical purposes
  - Must have access to detail information when needed

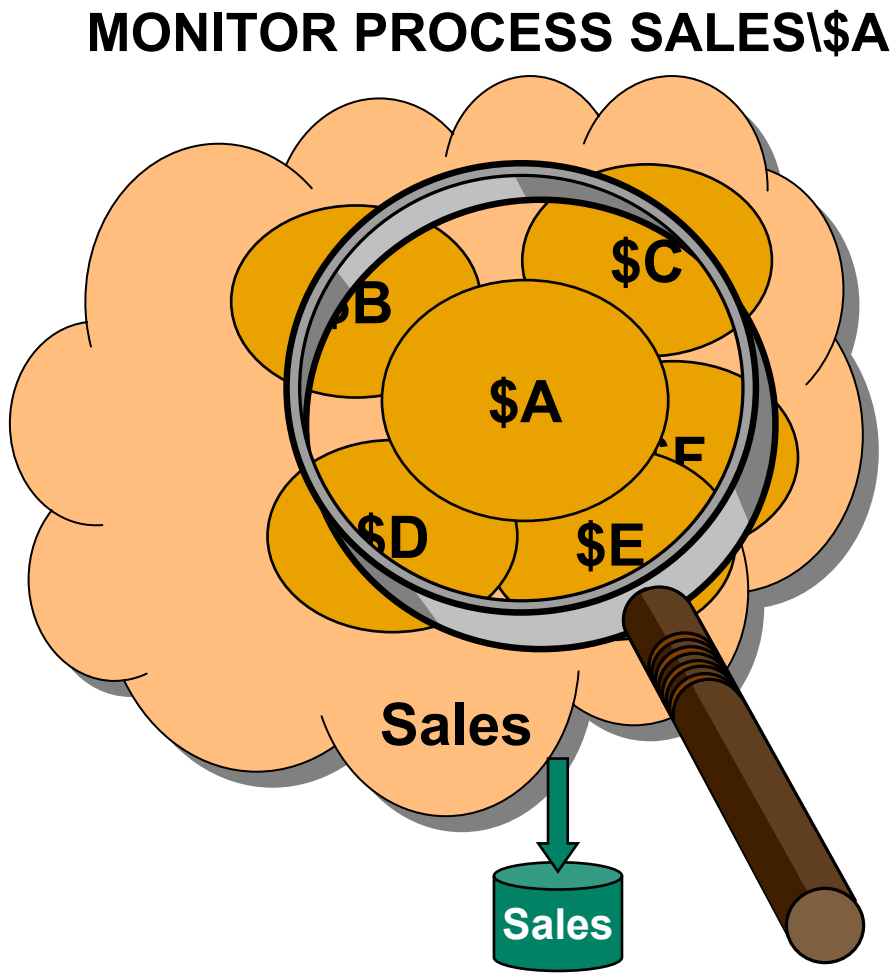
ASAP Process Monitoring

- ASAP
  - examines each individual monitored process at each interval
  - computes and analyzes over two dozen attributes for each process
  - compares each state-pair attribute against pre-defined objectives
  - generates EMS and/or ASAP alerts when objectives aren't met
  - stores historical information about each process in the ASAP database
  - provides alert, status and/or performance data to a variety of optional user-interfaces.



ASAP 2.4 – Process ZLE

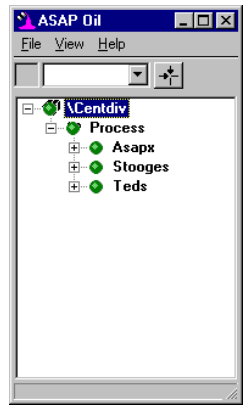
- Hierarchical Process Grouping
  - Allows logical groups of processes
  - For example group all servers for the sales application
  - Or group all spooler processes
- Aggregation/Propagation to Group Levels
  - Provides aggregate summaries and alerts
  - Real-time historical data reduction
- Configuration Options
  - 5 Levels of hierarchical naming
  - Aggregate domains at any level
  - Control of aggregate/propagate function



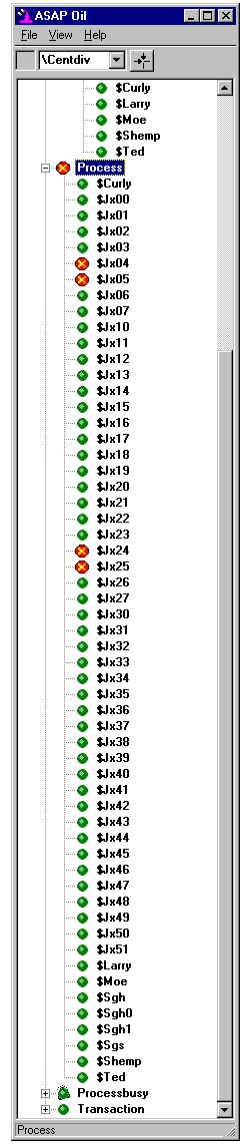
ASAP 2.4 - Hierarchical Process Grouping

- The MONITOR Command Defines Process Domains
  - MONITOR PROCESS $ABC
  - MONITOR PROCESS SALES\ORDERS\SERVERS\$ABC
  - MONITOR PROCESS SPOOLER\SUPERVISORS\$SPLS

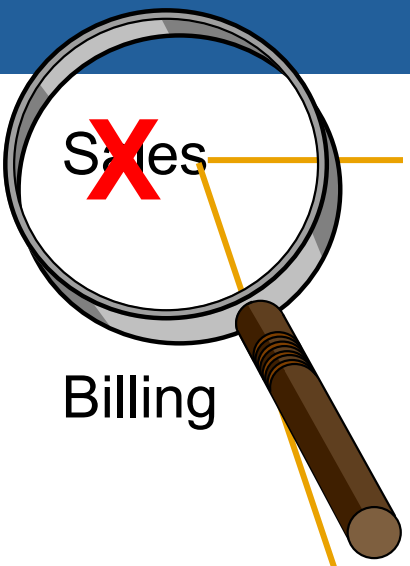
- Benefits of Process Groups
  - Hierarchical views and state propagation
  - Aggregate summaries at each level
  - More manageable navigation and process monitoring
  - Monitor the service and the individual components



- Process Domain Name Rules
  - Up to 64 bytes in length
  - Up to 5 hierarchical levels, separated by a backslash (“\”)
  - Last level denotes process name in abstract group



ASAP 2.4 - Hierarchical Process Naming



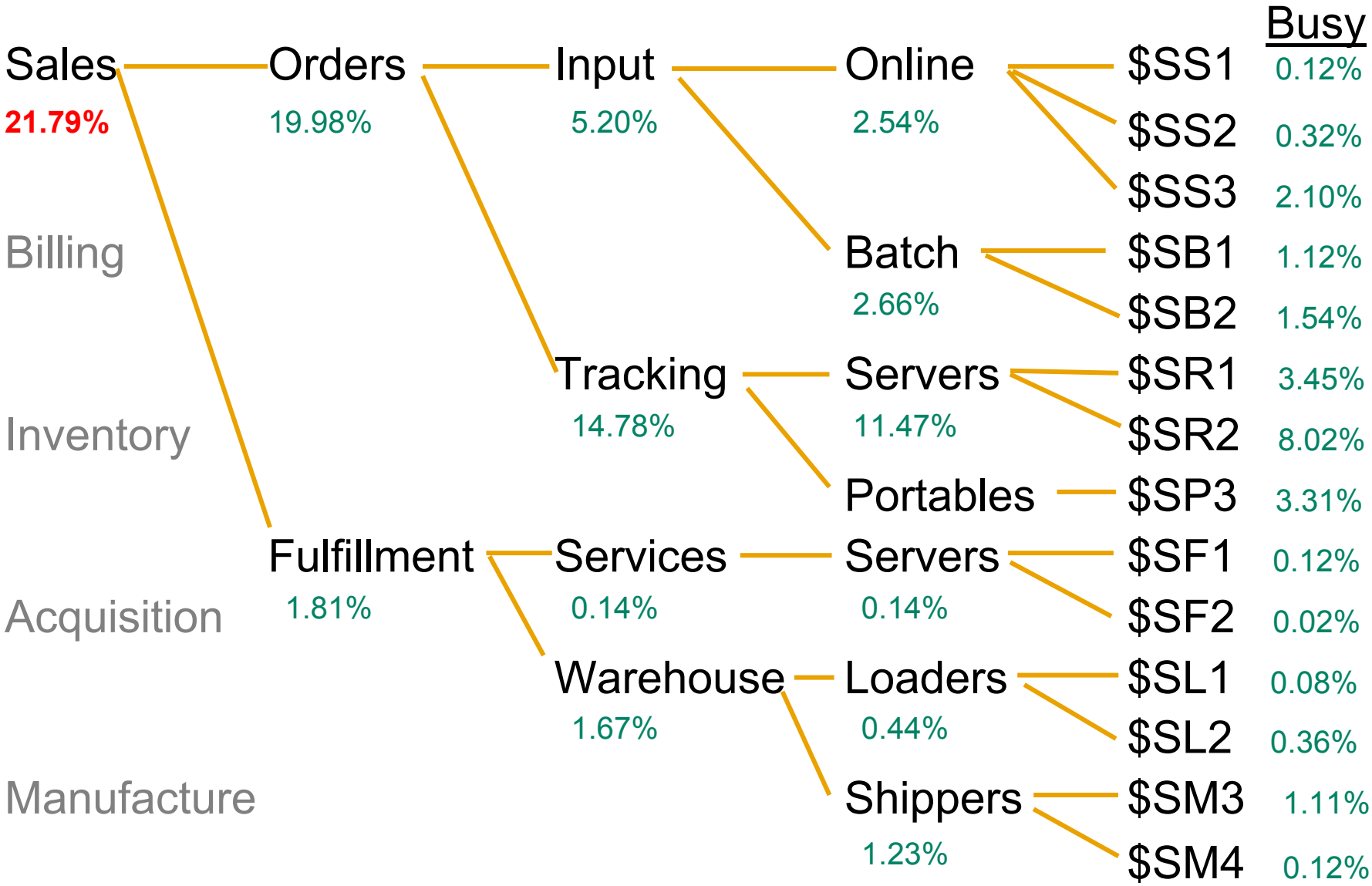
ASAP 2.4 – Creating Aggregate Domains

- Use the MONITOR command to create aggregate domains
  - MONITOR PROCESS SALES\ORDERS\SERVERS\#
  - MONITOR PROCESS SALES\ORDERS\#
  - MONITOR PROCESS SALES\#
- Each MONITOR PROCESS <name>\# command instructs ASAP to create an aggregate record of that name
- The aggregate record contains attribute values that are computed across all group members at and below that level in the name hierarchy
- Attribute values in the summary records are determined by Data Aggregation or Attribute Propagation
- Attribute states in the summary records are determined using State Propagation or an objective comparison

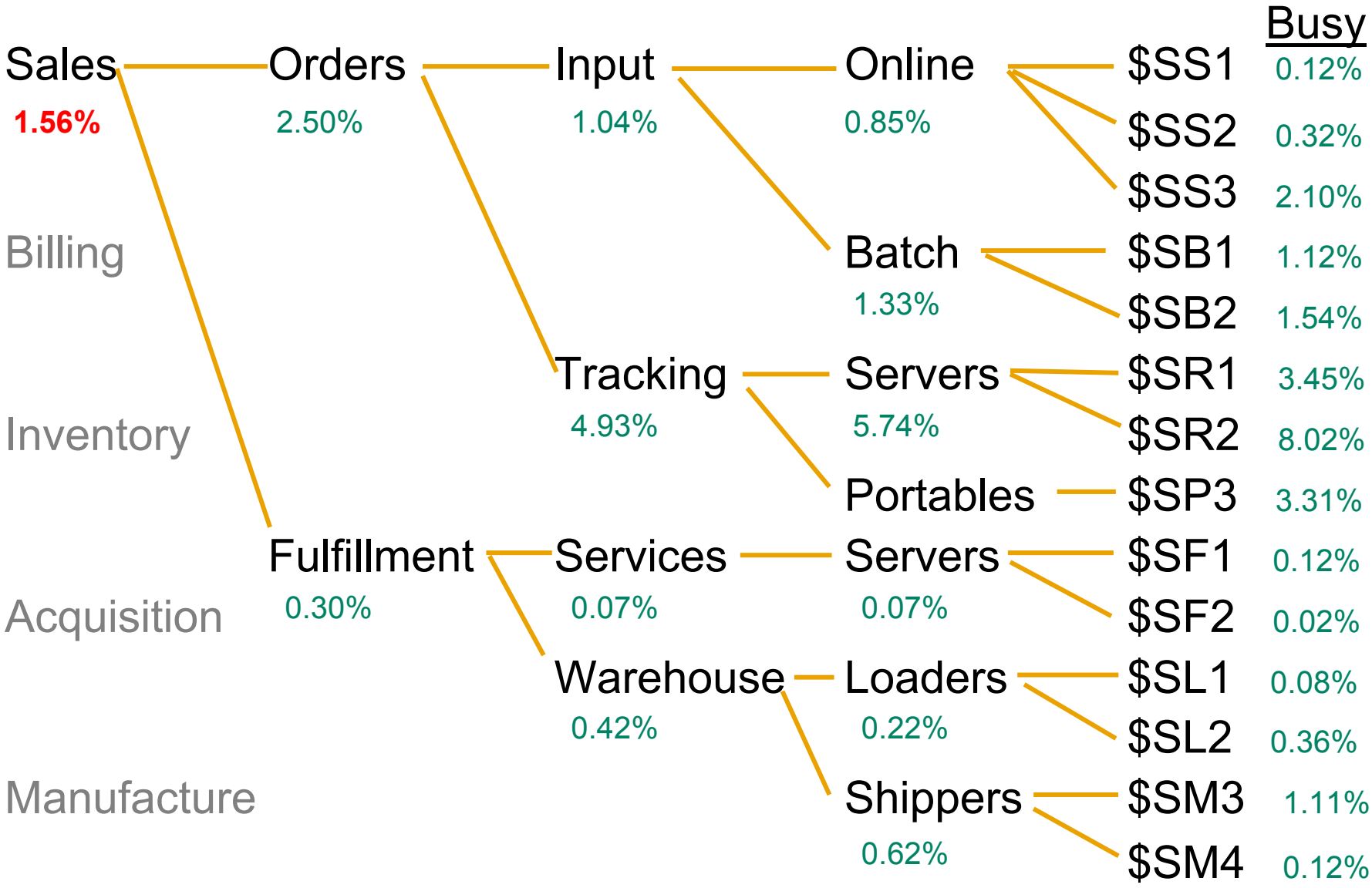
ASAP 2.4 - Aggregation and Propagation

- Data Aggregation
  - ASAP combines attribute values using a mathematical comparison or computation across all members of the group
  - ASAP uses SUM, AVG, MIN, MAX or CNT functions to determine the attribute value in the aggregate record
- Attribute Propagation
  - If no Data Aggregation is specified for an attribute ASAP propagates an attribute value to the aggregate record
  - The attribute value with the worst (highest) ASAP state gets propagated to the aggregate record
- State Propagation
  - ASAP propagates the worst (highest) ASAP state to the aggregate record unless an objective is set on the aggregate domain attribute

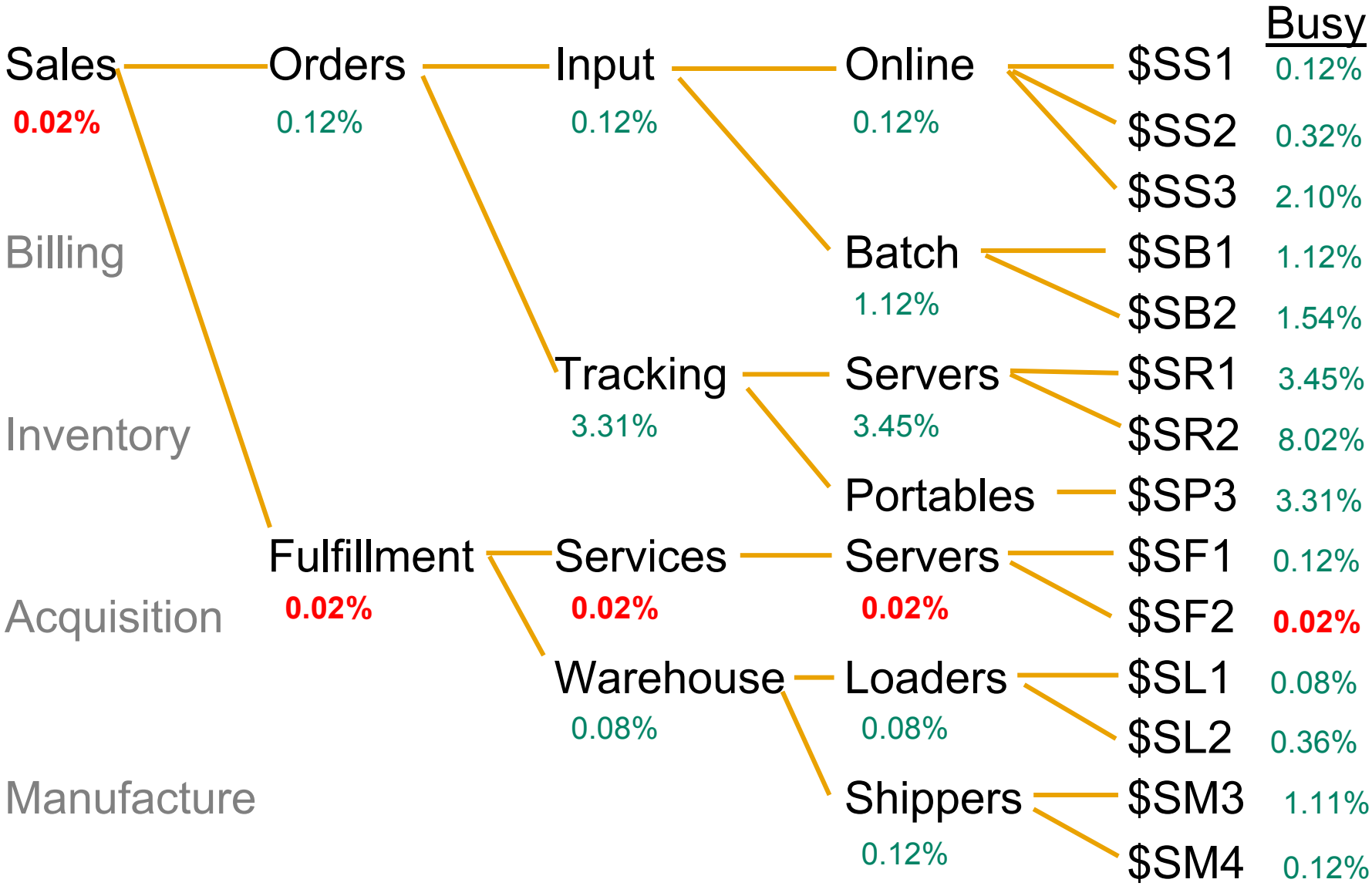
ASAP 2.4 - Busy Attribute - Aggregate SUM



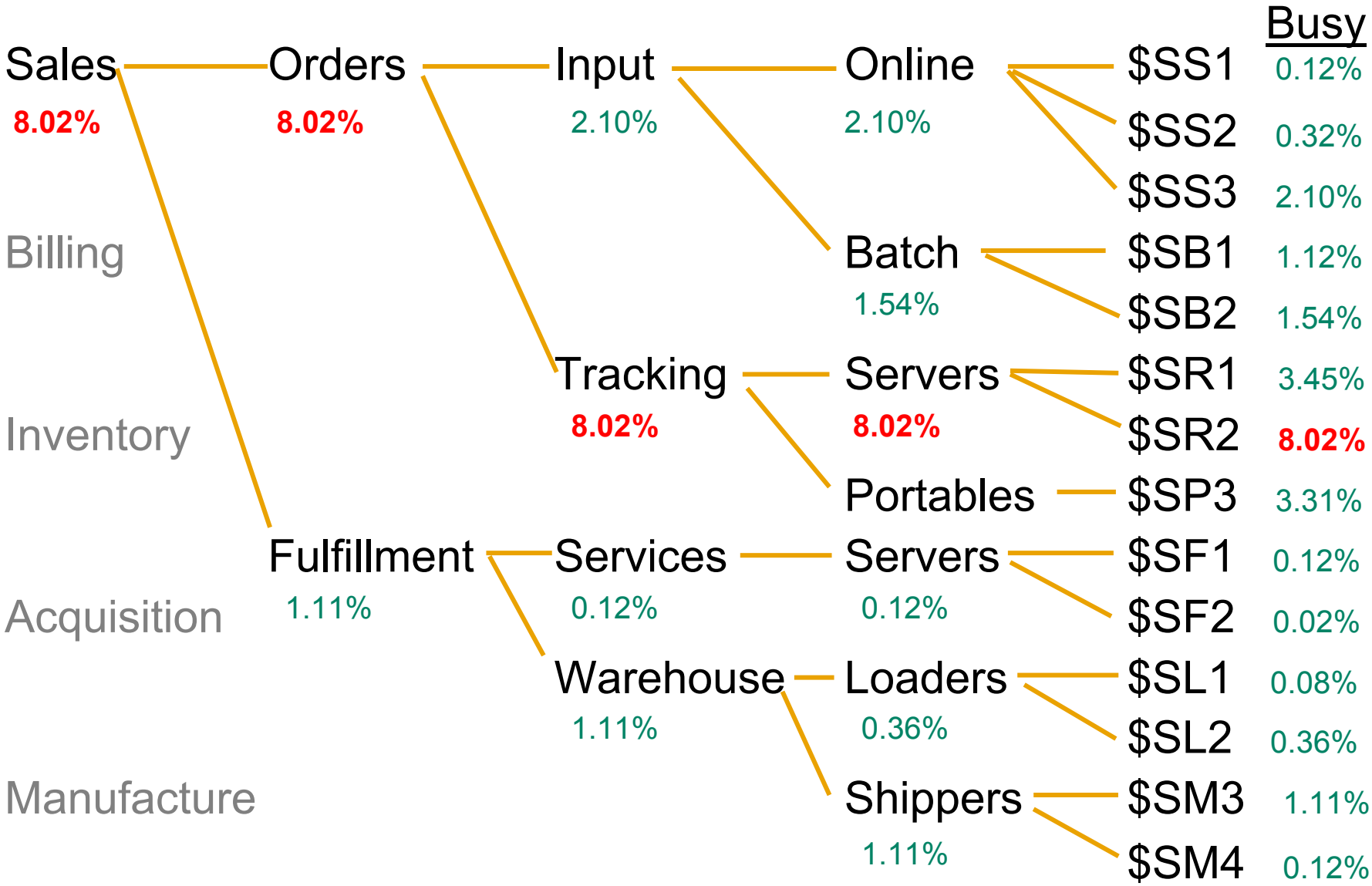
ASAP 2.4 - Busy Attribute - Aggregate AVG



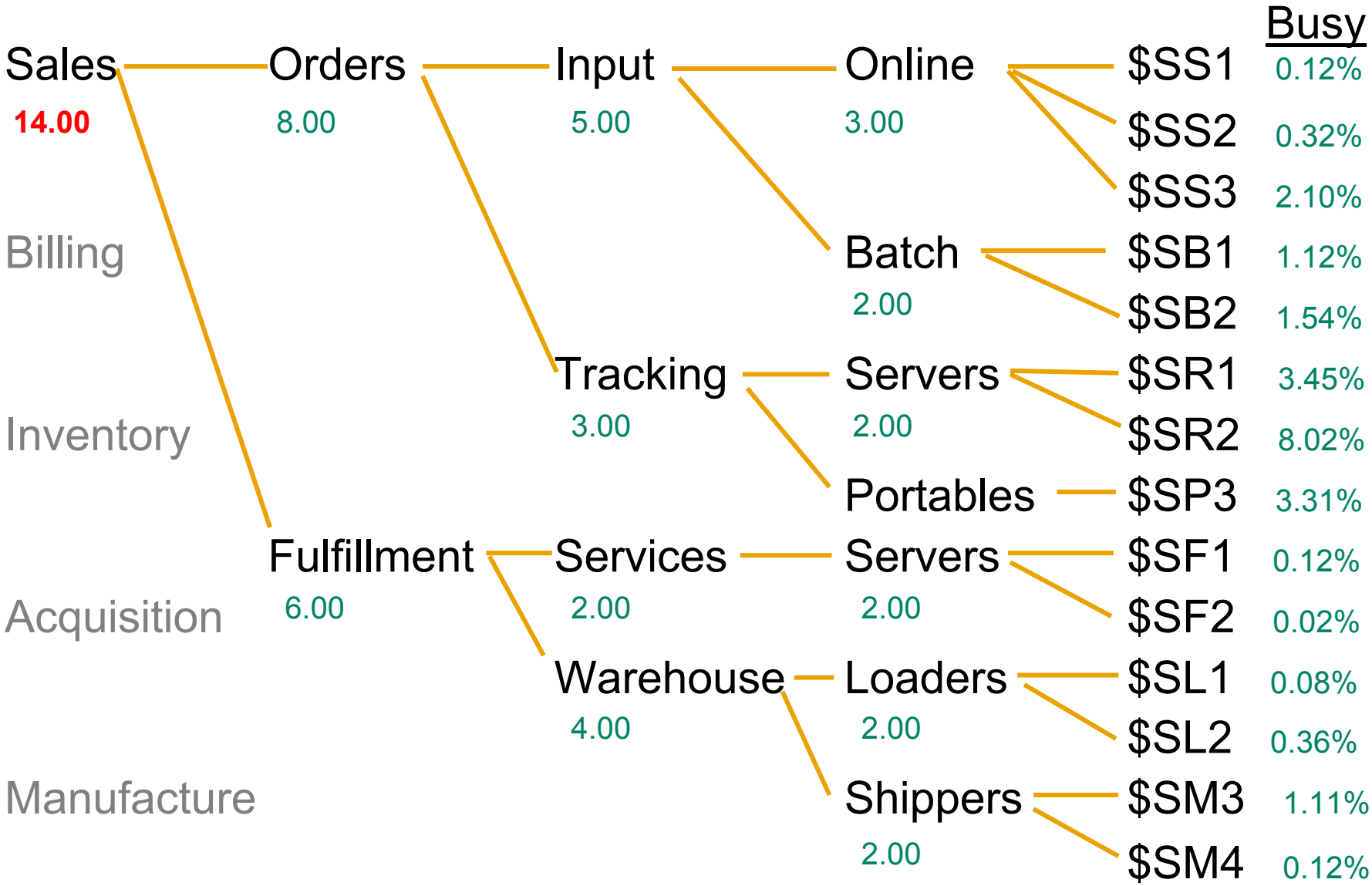
ASAP 2.4 - Busy Attribute - Aggregate MIN



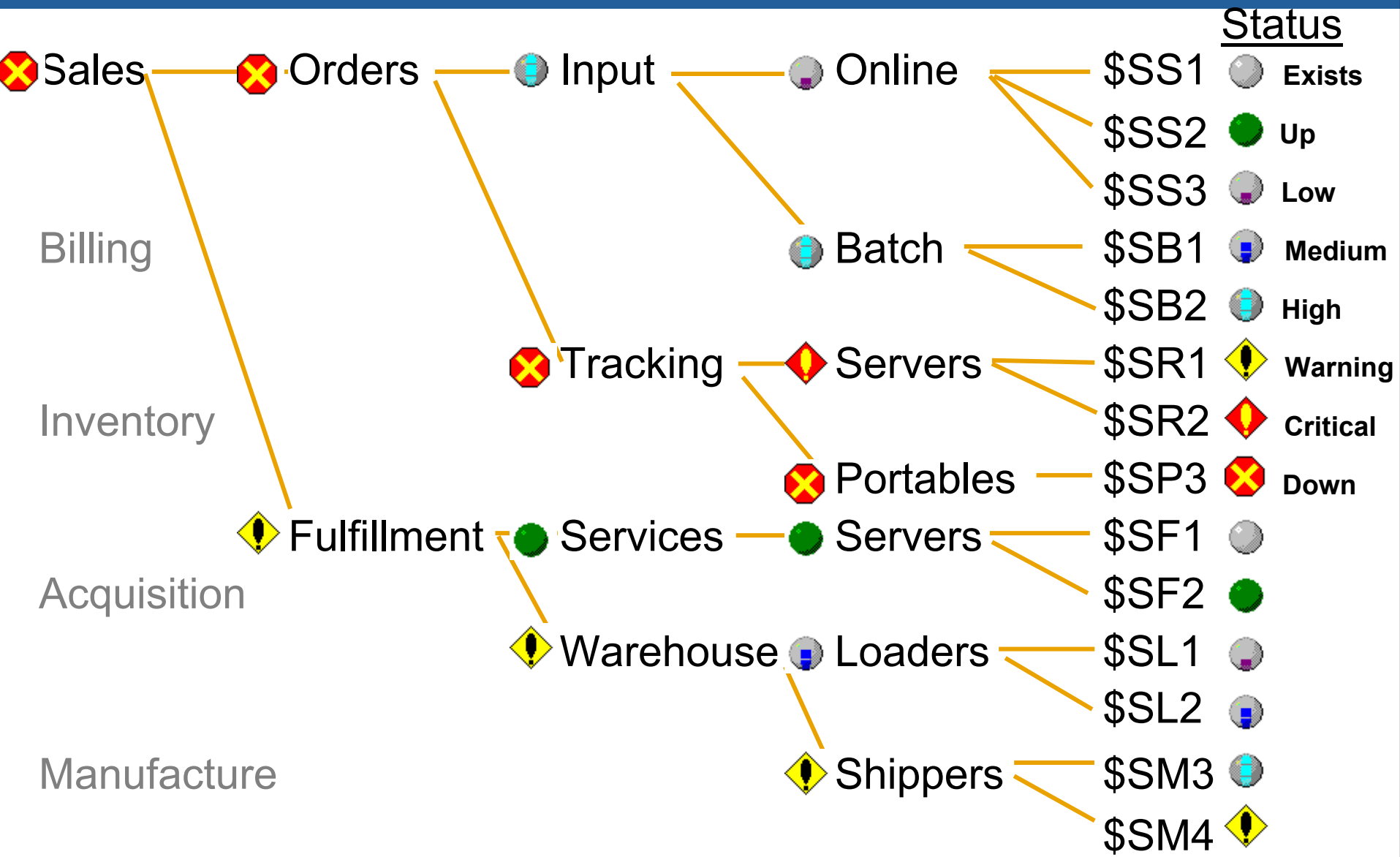
ASAP 2.4 - Busy Attribute - Aggregate MAX



ASAP 2.4 - Busy Attribute - Aggregate CNT



ASAP 2.4 - Propagation using ASAP State



ASAP 2.4 - Process Aggregate Record

Domain	Status	Op	Time	Error	S1	Cpu	S2	Pri	S3	Busy	S5	PState	S6
Sales\#	4 (3) Processes	8	09:00	0	1	4	7	10	7	0.07	2	Unalloc	1
Sales\$A	Up	2	09:00	0	1	2	2	10	2	0.01	2	Runnable	1
Sales\$B	Up	2	09:00	0	1	1	2	180	2	0.01	7	Runnable	1
Sales\$C	Up	2	09:00	0	1	4	7	165	7	0.01	2	Runnable	1
Sales\$D	Up	2	09:00	0	1	3	2	180	2	0.01	2	Runnable	1
Sales\$E	Down	8	09:00	0	1	1	2	180	2	0.01	2	Unalloc	1
Sales\$F	Up	2	09:00	0	1	7	2	180	2	0.01	2	Suspend	1
Sales\$G	Up	2	09:00	0	1	6	2	180	2	0.01	2	Runnable	1

```
RANK PROCESS SALES,      CPU   <> 4,    PRI >= 180
RANK PROCESS SALES\$A, PRI     <= 10
RANK PROCESS SALES\$B, BUSY   > .05 CRITICAL REPEAT
RANK SALES\#, BUSY < 1, BUSY > .05
```

ASAP 2.4 - Process Aggregate Record

Domain	Status	Time	Error	Cpu	Pri	Busy	PState
Sales\#	4 (3) Processes	09:00	0	4	10	0.07	Unalloc
Sales\$A	Up	09:00	0	2	10	0.01	Runnable
Sales\$B	Up	09:00	0	1	180	0.01	Runnable
Sales\$C	Up	09:00	0	4	165	0.01	Runnable
Sales\$D	Up	09:00	0	3	180	0.01	Runnable
Sales\$E	Down	09:00	0	1	180	0.01	Unalloc
Sales\$F	Up	09:00	0	7	180	0.01	Suspend
Sales\$G	Up	09:00	0	6	180	0.01	Runnable

RANK PROCESS SALES, CPU <> 4, PRI >= 180
RANK PROCESS SALES\$A, PRI <= 10
RANK PROCESS SALES\$B, BUSY > .05 CRITICAL REPEAT
RANK SALES\#, BUSY < .1, BUSY > .04

ASAP 2.4 - Aggregate Only Domains

- The MONITOR Command Defines Aggregate Only Domains
  - MONITOR PROCESS SALES\ORDERS\##
  - MONITOR PROCESS SALES\##
- Aggregate Only
  - Process each detail record
  - Combine data into an aggregate record
  - Write **ONLY** the aggregate record to the ASAP database
  - Store the data from hundreds or thousands of processes in a single historical record
- Detail Records Can Still be Seen !
  - PROCESS, MEMORY option

ASAP 2.4 - Process Command Options

- AGGREGATE and AGGREGATEONLY
  - Use AGGREGATE to display aggregate process domains
  - Use AGGREGATEONLY | AGGONLY | AO to display only aggregate domains
- MEMORY
  - Retrieves detail information directly from ASAP memory
  - Enables an ASAP Client user to right-mouse on an alerting aggregate only domain to see the detail domains that are causing the alert
- MINSTATE [AUTO | <ASAP state>]
  - Extended to apply to domains in memory
- COUNT <n>
  - Use to limit the response back to ASAP Client

ASAP 2.4 - SET Process Param Options

- MINSTATE <ASAP state>
  - Defines the state at which a Process domain is considered to be alerting
  - Default is 3, Low Alert
- AGG and AGGONLY
  - Automatically creates aggregate or aggregate only domains at all levels of all process hierarchies
  - MONITOR commands are not required
- MINONLY
  - Exception mode processing where only alerting detail domains are written to the ASAP database
- NORECS
  - Write no records to the ASAP historical database (OMF mode)

ASAP 2.4 - More SET Options

- SET MONITORADD ON | OFF
  - ON means the behavior for “MONITOR <entity> <object>” is to ADD the domain to set of domains to be monitored
  - OFF means an explicit “ADD” must be specified in order to add a domain to the set of monitored domains
  - Default is MONITORADD ON
- SET OBJECTIVESEVENTSUBJECT ON | OFF
  - ON means the domain name token (ZASP-TKN-ASAP-DNAME) will be used as the event subject in ASAP alert events
  - OFF means the ASAP component token (ZASP-TKN-ASAP-COMPONENT) will be used as the event subject in ASAP alerts

ASAP Extension 2.4

Can you monitor? 10,000 application domains in a processor?

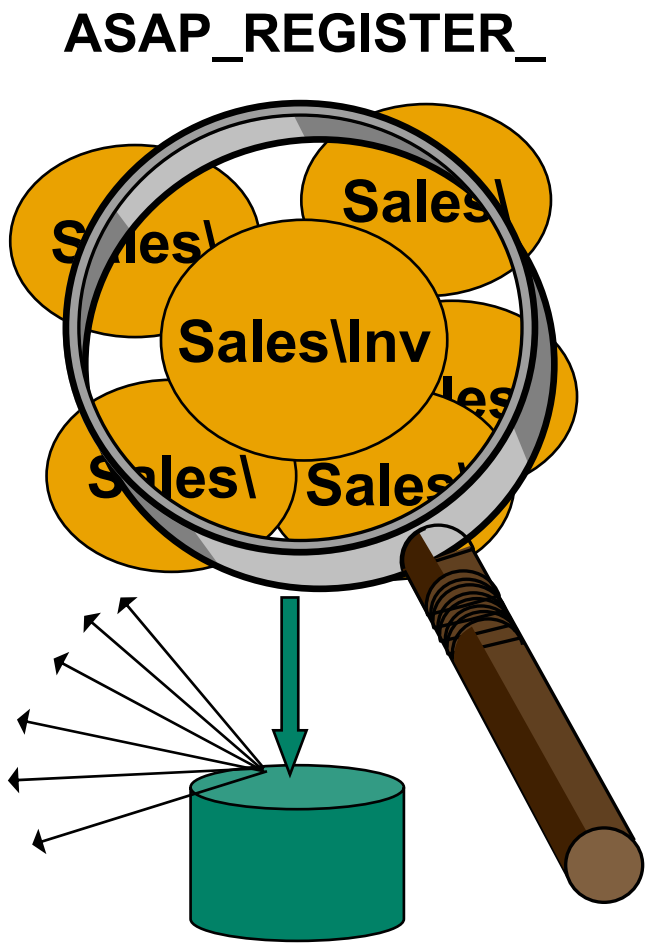
160,000 domains in a system?

40,800,000 domains in an Expand network?

- That's what's here today in ASAPX 2.4!
- Requirements:
  - Must analyze each individual domain and compare against predefined service levels, alerting when necessary
  - Must be able to group domains and view at the group level
  - Must be able to aggregate/propagate data and alerts to the group level
  - Must be able to set service level objectives on aggregate data
  - Must be able to store summary information about the group for historical purposes
  - Must have access to detail information when needed

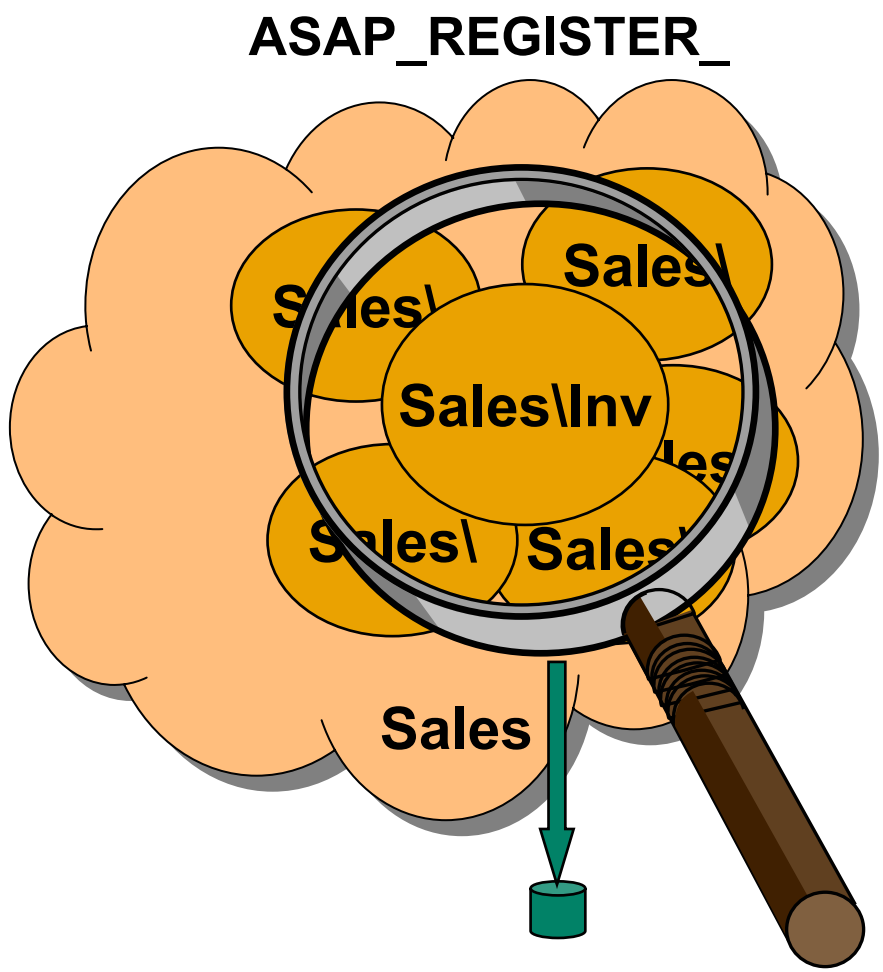
ASAP Extension - Domain Monitoring

- ASAPX
  - examines each registered domain at each interval
  - computes or determines from 1-30 attributes for each domain
  - compares each state-pair attribute against pre-defined objectives
  - generates EMS and/or ASAP alerts when objectives aren't met
  - stores historical information about each process in the ASAP database
  - provides alert, status and/or performance data to a variety of optional user-interfaces.



ASAP Extension 2.4 – ASAPX ZLE

- Hierarchical Process Grouping
  - Supported since ASAP 1.0
- Aggregation/Propagation to Group Levels
  - Supported since ASAP 2.0
- 10,000 domains per CPU
- 10,000 aggregates per CPU
- Control of aggregate function
- Detail record retrieval directly from ASAPX memory



ASAPX 2.4 – Max Domains and Aggregates

- SET MAXDOMAINS <number>
  - 1,024 – 10,000 domains per processor
  - Default is 1,024
  - Determines the size of the application shared-memory segment
  - Formula for size is: 10 + (244 * <number>)
  - Be careful if you are changing this value!

- SET MAXAGGREGATES <number>
  - Previously auto-set by ASAPX to 1x domains
  - 1,024 – 10,000 aggregate domains per processor
  - Default is 1,024
  - Determines the size of internal memory ASAPX allocates to store aggregates

ASAPX 2.4 – Aggregation Control

- DataItem and S operands now have optional aggregate codes
  - #<n> [SUM | AVG | MIN | MAX | CNT]
  - S [SUM | AVG]

- Example:
  - **DataItems “0 I”**
    - Item is defined as a count of successful transactions
    - ASAP_Update_(handle,errordtl,0,1) called after each EndTransaction
  - **MetricRule “#0/S”**
    - Produces a Transaction Rate, transactions per second
    - By default #0 is summed across all domains for the aggregate computation
    - S is taken as the average of S because #0 is an integer

ASAPX 2.4 – Aggregation Control

- MetricRule “#0/S”
  - Means MetricRule “#0 SUM / S AVG”
  - The default aggregate is therefore the overall transaction rate per second
- MetricRule “#0 AVG / S”
  - The aggregate is the average transaction rate
- MetricRule “#0 MAX / S”
  - The aggregate is the maximum transaction rate found among all the members of the group
- MetricRule “#0 MIN / S”
  - The aggregate is the minimum transaction rate found among all members of the group

ASAPX 2.4 –APP Command Options

- MEMORY
  - Retrieves detail data from memory for aggregate only domains
  - Use in conjunction with MINSTATE and COUNT options
  - Use in EDL Detail command for user-defined entities
  - Detail “APP ^,DE,ST,MEM,MIN,CO 20”
- EXTRACT <filename>
  - Extracts selected data from DBAPP to a new database file
  - New file is auto-created if it doesn’t exist
- NOPRINT
  - Use with EXTRACT to inhibit output to terminal
- NOLINEBREAK
  - Use when outputting to CSV or process that can handle lines longer than 132 characters
- AGGONLY, AO (abbreviations for AGGREGATEONLY)

ASAPX 2.4 –Last but not Least

- New ASAPX Dynamic Link Library
  - ASAPXDLL
  - Compiled from same code base as all other libraries
  - Use with PIC applications
- New DataItem Type
  - DATAITEM “<n> T”
  - Specifies DataItem is 8-byte ASCII text string
  - Should use MATH 2 (replace text)
  - Should **not** use AVG or SUM aggregate controls

NonStopAsap.com