What we do with Go

agenda

- how we started with Go
- RichClient/JVM Distribution
- Business Event Monitoring
- using Go, why Go is Fun

a few problems solved

- 2014 RichClient/JVM distribution (groupcache)
- 2014 Business Event Monitoring with time series (d3/cubism)
- 2014 Business Operation Monitor (ng)
- 2015 OCR Solution (omnipage, rewritten (old) C++ server)

how we started with Go

- Insurance Sector
- Windows “Servers”, MSSQL, Winddows File Servers, ActiveDirectories…
- but JEE/JBoss Java AppServer
- eclipse RCP based RichClient, about 200 users
- multiple generations (32bit/64bit, old OCR 32bit Solution)

SOAPCaller

- “new” external SOAP Service
- Monitoring: no SOAP support
- external sensors via exit code and stdout
- *.bat, java app, Windows-Script… => why not Go

SOAPCaller

- SOAPCaller
  - send SOAP request
  - receive and parse SOAP response (parse XML)
  - report via exit code & stdout
- => one exe-File, starts & runs in 200ms, simple deployment

network issues (1)

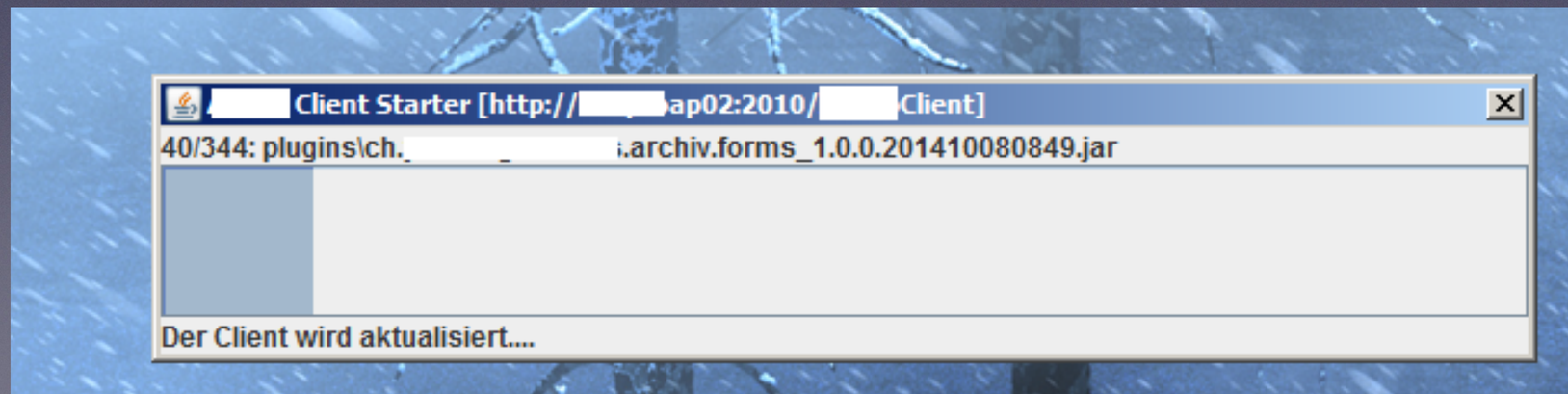
- new infrastructure (network and servers)
- RichClient App running on SMB Share
- ClassNotFound Exceptions in random order
- unreliable Network (L3 switches, ProxyARP issues)

network issues (2)

- “the network is slow”, “something is wrong”
- blocking Clients
- business processes unavailable / blocking
- printers unavailable 25% of the time => L3 Switches

RichClient/JVM distribution

- Windows SW distribution not yet available
- don’t run apps on “shares”
- distribute them to the client
- update, staging and multiple locations



Indexing Server

- walk over a given directory by configuration (JSON)
- read files into memory, SHA1 hash them and put them into a map
- combine configuration and the file index into a boot file
- provide the boot file over HTTP

bootManifest_peer.json

```
{
  "JavaOptions": {
    "Xmx": "512M",
    "MinVersion": "1.6",
    "MaxVersion": "1.6"
  },
  "ServerOptions": {
    "Path": "C:\\distr\\20140415\\",
    "Piggyback": "",
    "Port": "2009",

    "GroupType": "Peer", // Alone, Peer, ProxyPeer
    "GroupName": "xxx-ACC-Test",
    "GroupCacheMe": "http://10.9.9.139:50001",
    "GroupCachePeers": ["http://10.9.9.139:50001"],
    "GroupCacheSize": "512MiB", // 512MiB, 1GiB

    "Debug": true
  },
  "ClientOptions": {
    "KillAllRunningProcesses": false,
    "WelcomeMessage": "Heute Abend ab 1700 wird ein Hotfix eingespielt. Bitte den Client beenden.",
    "UploadScreenshots": false,
    "Path": "\\Ax\\Ax-Prod\\",
    "Version": "20140415T1352_2",
    "Exec": "\\Ax.exe",
    "Args": [
      "-vm C:\\Apps\\Ax\\Ax-Prod\\20140415T1352_2\\jre6\\bin\\javaw.exe"
    ],
    "KeepLastVersions": 2,
    "UITitle": "Ax Client Starter"
  },
  "ClientFiles": []
}
```

```
type ClientFile struct {
    Version  string
    FileName string
    Sha1     string
}
```

ClientStarter (1)

- ClientStarter.exe + server.json (JAVA Swing App, launch4j EXE)
- on every start
  - download boot.json from Index Server
  - synchronize index files locally
  - boot the app

ClientStarter (2)

- a tuesday afternoon
- worked more or less in second try
- structs, JSON en-/decoding, filepath.Walk(), maps, “net/http”

Issues

- after a few hours, server crashed, dump in windows memory allocation/management
- debug.WriteHeapDump(f.Fd())
- https://golang.org/issue/8119 (solved within 2 days)
- => switched to Windows 64Bit

acs_crash_13b2_04.txt

```
fatal error: runtime: cannot map pages in arena address space

goroutine 40 [running]:
runtime.throw(0x8391c3)
        c:/go/src/pkg/runtime/panic.c:520 +0x71 fp=0x42de362c
runtime.SysMap(0x0, 0x4000000, 0x42de3601, 0x841418)
        c:/go/src/pkg/runtime/mem_windows.c:117 +0x96 fp=0x42de364c
runtime.MHeap_SysAlloc(0x84bec0, 0x4000000)
        c:/go/src/pkg/runtime/malloc.goc:615 +0xf7 fp=0x42de367c
MHeap_Grow(0x84bec0, 0x2000)
        c:/go/src/pkg/runtime/mheap.c:319 +0x57 fp=0x42de36a4
MHeap_AllocLocked(0x84bec0, 0x2000, 0x0)
        c:/go/src/pkg/runtime/mheap.c:222 +0x2f4 fp=0x42de36c4
runtime.MHeap_Alloc(0x84bec0, 0x2000, 0x0, 0x5e8e0101)
        c:/go/src/pkg/runtime/mheap.c:178 +0x8b fp=0x42de36d8
largealloc(0x1, 0x42de3730)
        c:/go/src/pkg/runtime/malloc.goc:223 +0x97 fp=0x42de36fc
runtime.mallocgc(0x3fffe00, 0x60aca1, 0x1)
        c:/go/src/pkg/runtime/malloc.goc:168 +0xbc fp=0x42de3730
cnew(0x60aca0, 0x3fffe00, 0x1)
        c:/go/src/pkg/runtime/malloc.goc:835 +0xad fp=0x42de3740
runtime.cnewarray(0x60aca0, 0x3fffe00)
        c:/go/src/pkg/runtime/malloc.goc:848 +0x3f fp=0x42de3750
makeslice1(0x600ee0, 0x3fffe00, 0x3fffe00, 0x42de3790)
        c:/go/src/pkg/runtime/slice.goc:55 +0x4b fp=0x42de375c
runtime.makeslice(0x600ee0, 0x3fffe00, 0x0, 0x3fffe00, 0x0, 0x0, 0x3fffe00, 0x3fffe00)
        c:/go/src/pkg/runtime/slice.goc:36 +0xb2 fp=0x42de377c
bytes.makeSlice(0x3fffe00, 0x0, 0x0, 0x0)
        c:/go/src/pkg/bytes/buffer.go:191 +0x72 fp=0x42de37a0
bytes.(*Buffer).ReadFrom(0x4760aae0, 0x2019a8, 0x475af530, 0x1fffe00, 0x0, 0x12787a00, 0x0)
        c:/go/src/pkg/bytes/buffer.go:163 +0xa4 fp=0x42de3814
io/ioutil.readAll(0x2019a8, 0x475af530, 0x200, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0)
        c:/go/src/pkg/io/ioutil/ioutil.go:33 +0x135 fp=0x42de3868
io/ioutil.ReadAll(0x2019a8, 0x475af530, 0x475af530, 0x2019a8, 0x475af530, 0x0, 0x48c8a690)
        c:/go/src/pkg/io/ioutil/ioutil.go:42 +0x46 fp=0x42de3890
[redacted]er/src/http.(*httpGetter).Get(0x47a52250, 0x0, 0x0, 0x47b0ab20, 0x47b0ab40, 0x0, 0x0)
[redacted]oStarter/src/http/http.go:220 +0x4d5 fp=0x42de3950
[redacted]0x1275e000, 0x0, 0x0, 0x201888, 0x47a52250, 0x475af410, 0x30, 0x0, 0x0, 0x0, ...)
[redacted]oStarter/src/github.com/golang/groupcache/groupcache.go:275 +0xea fp=0x42de39cc
[redacted]0x16e0b2a0, 0x42de3ab0, 0x42de3aac)
[redacted]oStarter/src/github.com/golang/groupcache/groupcache.go:233 +0xfb fp=0x42de3a80
[redacted].Do(0x1275e0a4, 0x475af410, 0x30, 0x42de3ae4, 0x4ceb81, 0x1275e068, 0x475af410, 0x30)
[redacted]oStarter/src/github.com/golang/groupcache/singleflight/singleflight.go:56 +0x18f fp=0x42de3abc
[redacted]000, 0x0, 0x0, 0x475af410, 0x30, 0x201848, 0x47b0ab00, 0x0, 0x0, 0x0, ...)
[redacted]oStarter/src/github.com/golang/groupcache/groupcache.go:254 +0xcf fp=0x42de3b00
[redacted]00, 0x0, 0x0, 0x475af410, 0x30, 0x201848, 0x47b0ab00, 0x30, 0x12c7a160)
[redacted]oStarter/src/github.com/golang/groupcache/groupcache.go:215 +0x1f4 fp=0x42de3b78
[redacted]er/src/server.(*Server).readFromProxyCache(0x12720540, 0x12707500, 0x4760a8c3, 0x28, 0x0, 0x0, 0x3, 0x41fc2b, 0x42de3ca0)
[redacted]oStarter/src/server/server.go:264 +0x161 fp=0x42de3c4c
[redacted]er/src/server.(*Server).groupHandler(0x12720540, 0x201810, 0x48c8a620, 0x48c8a540)
```

ClientStarter 2.0

- multiple remote locations (DE, IT, FR)
- staging, jre6

➡ GroupCache (github.com/golang/groupcache) (small patch for server selection)

➡ expiring keys for boot.json => boot.json_20140701T165_

➡ took another afternoon (+ a bit more to polish)

➡ runs over a year without issues

bootManifest_pp.json

```
{
  "ServerOptions": {
    "Port": "2010",
    "Debug": true,

    "GroupType": "ProxyPeer", // Alone, Peer, ProxyPeer
    "GroupCacheMe": "http://10.9.9.139:50001",
    "JoinCaches": false,
    "GroupCacheProxyPeers":
        [
        {
          "GroupName": "Ax-ACC",
          "GroupType": "ProxyPeer",
          "GroupCachePeers": ["http://192.168.30.17:50002"],
          "GroupCacheSize": "1GiB"
        },
        {
          "GroupName": "Ax-PRO",
          "GroupType": "ProxyPeer",
          "GroupCachePeers": ["http://192.168.30.25:50002"],
          "GroupCacheSize": "1GiB"
        },
        {
          "GroupName": "Bx-ACC",
          "GroupType": "ProxyPeer",
          "GroupCachePeers": ["http://192.168.30.33:50002"],
          "GroupCacheSize": "1GiB"
        },
        {
          "GroupName": "Bx-PRO",
          "GroupType": "ProxyPeer",
          "GroupCachePeers": ["http://192.168.30.34:50002"],
          "GroupCacheSize": "1GiB"
        },
        {
          "GroupName": "Cx-PRO",
          "GroupType": "ProxyPeer",
          "GroupCachePeers": ["http://192.168.30.31:50002"],
          "GroupCacheSize": "1GiB"
```

Business Event Monitoring with time series

“A time series is a sequence of data points, typically consisting of successive measurements made over a time interval.”
[wikipedia]

http://d3js.org
https://square.github.io/cubism

Quotes from Business Support

- “its slow”
- “at 9am, we had a lot of blocking clients”
- “something is wrong”
- “we are missing documents in ‘Postfach’”

what do you see ?

PRTG Network Monitor - prtg.skigh.com (prtgadmin) SSL

File Edit View Help

PRTG Network Monitor

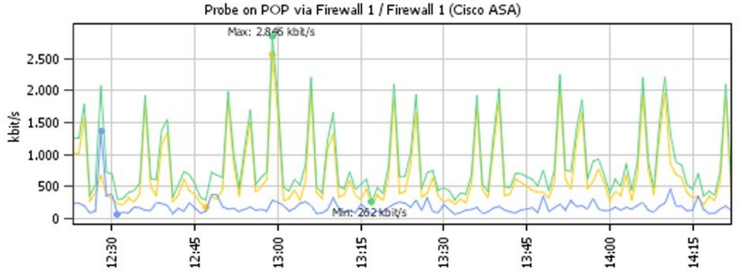
6 692 6 4

Devices | Sensors | Alarms | Maps | Reports | Logs | Todos | Setup

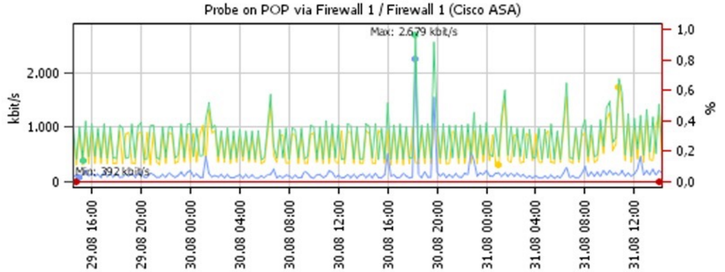
- Root
  - Probe on POP via Firewall 1
    - Local Probe on POP for prtg.sk 17
    - Paessler AG ** Mission Cri
      - Anbindung BSX via IPX
        - Erster Hop bei IPX hinte 2
        - SDSL Router 1 (Efficien 5
        - SDSL Router 2 (Efficien 4
        - DSL Load Balancer (Cisc 2
      - BSX Firewalls and Data
        - Firewall 1 (Cisco ASA) 8
          - PING 1 0 msec
          - CPU Load 1 2 %
          - DSL (ASA WAN Inte 553 kbit/s
          - ASA LAN Interface 507 kbit/s
          - SNMP Messages In 1 SNMP/s
          - SNMP Messages Ou 1 SNMP/s
          - Free Proc. Memory 59.477 kb
          - (002) Adaptive Sec 454 kbit/s
        - Firewall 2 (LANCOM) 19
        - Firewall 3 (pfSense) 4
      - BSX Backbone Server (1
      - Paessler AG External S 24
    - BSX Backoffice Systems (F 3
    - BSX Network (Switches & 1 1
    - Various Websites (via IPX) 9
  - Probe on Walldorf via Firewall
    - Probe Device 8
    - Monitoring for Paessler AG 7
    - Monitoring for beyond con
    - Various Websites (vianetw 7
  - Probe on Marvin via Firewall 3
    - Probe Device 7
    - Monitoring for Paessler AG 36
    - Various Websites (via VDS 7
  - Probe on NAS1
    - Probe Device 1 8
    - Load Test Device 3
  - Probe on NAS2
    - Probe Device 12
  - Probe on Virtuozzo
    - Probe Device 9

Overview | Graphs | Log | Settings | History

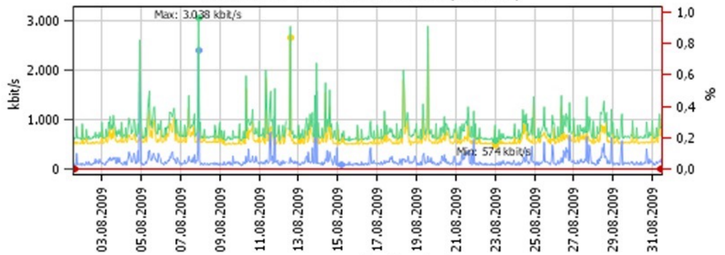
ASA LAN Interface (Live Graph, 2 hours)
Probe on POP via Firewall 1 / Firewall 1 (Cisco ASA)



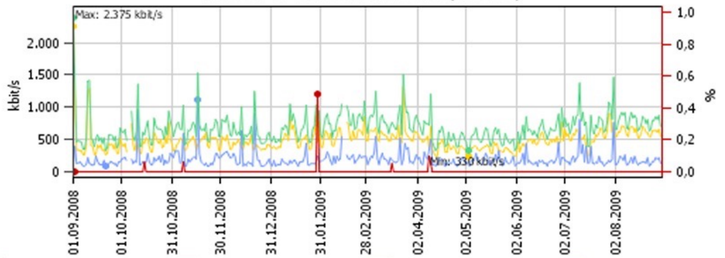
ASA LAN Interface (2 days)
Probe on POP via Firewall 1 / Firewall 1 (Cisco ASA)



ASA LAN Interface (30 days)
Probe on POP via Firewall 1 / Firewall 1 (Cisco ASA)



ASA LAN Interface (365 days)
Probe on POP via Firewall 1 / Firewall 1 (Cisco ASA)



Downtime (%) Sum (kbit/s) Traffic in (kbit/s) Traffic out (kbit/s)

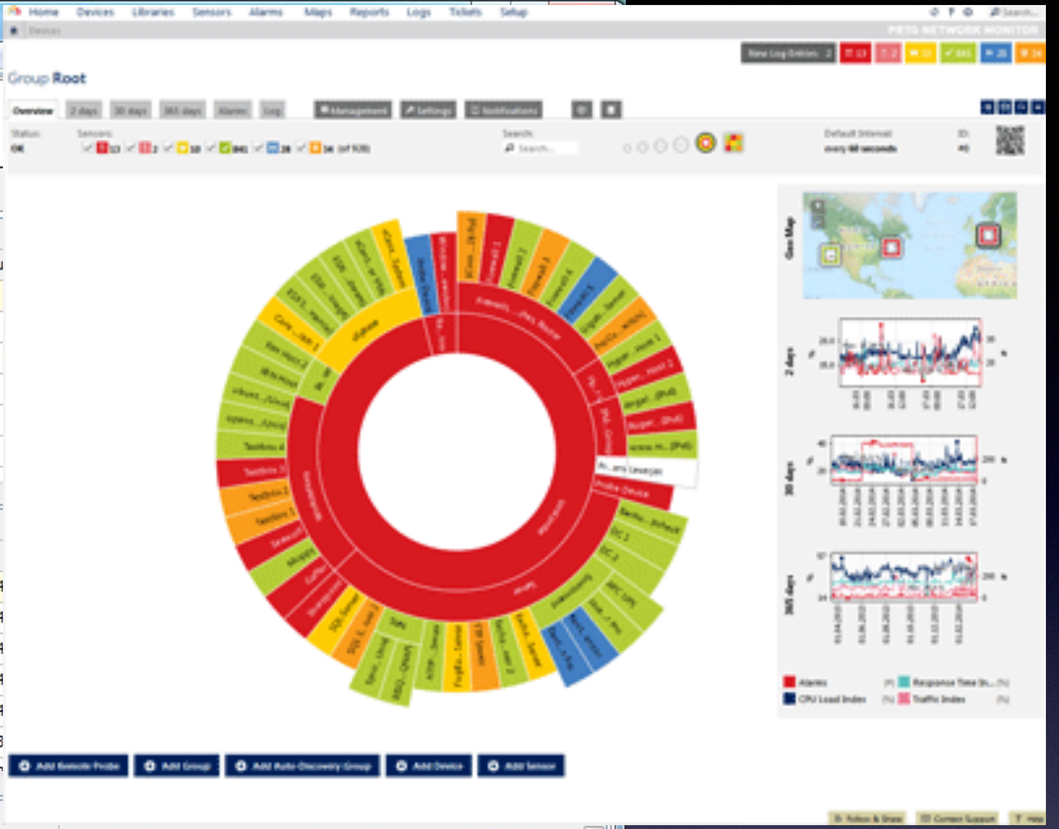
Date Time	Su
31.08.2009 14:22:47	
31.08.2009 14:21:47	
31.08.2009 14:20:47	
31.08.2009 14:19:47	
31.08.2009 14:18:47	
31.08.2009 14:17:47	

Date Time
31.08.2009 14:15:00 - 14
31.08.2009 14:10:00 - 14
31.08.2009 14:05:00 - 14
31.08.2009 14:00:00 - 14
31.08.2009 13:55:00 - 14
31.08.2009 13:50:00 - 13

Date Time	Sum (volume)	Sum (speed)	Traffic in (volume)	Traffic in (speed)	T
31.08.2009 13:00:00 - 14:00:00	354.763 KByte	807 kbit/s	72.721 KByte	165 kbit/s	
31.08.2009 12:00:00 - 13:00:00	380.093 KByte	865 kbit/s	84.574 KByte	192 kbit/s	
31.08.2009 11:00:00 - 12:00:00	359.290 KByte	818 kbit/s	62.927 KByte	143 kbit/s	
31.08.2009 10:00:00 - 11:00:00	489.676 KByte	1.114 kbit/s	67.994 KByte	155 kbit/s	
31.08.2009 09:00:00 - 10:00:00	400.831 KByte	912 kbit/s	66.647 KByte	152 kbit/s	
31.08.2009 08:00:00 - 09:00:00	294.135 KByte	669 kbit/s	56.121 KByte	128 kbit/s	

ASA I

Date Time	Sum (volume)	Sum (speed)	Traffic in (volume)	Traffic
30.08.2009 00:00:00 - 00:00:00	7.208.682 KByte	684 kbit/s	1.390.194 KByte	
29.08.2009 00:00:00 - 00:00:00	7.344.400 KByte	696 kbit/s	1.494.822 KByte	
28.08.2009 00:00:00 - 00:00:00	9.102.575 KByte	869 kbit/s	2.177.589 KByte	
27.08.2009 00:00:00 - 00:00:00	8.534.921 KByte	809 kbit/s	1.981.302 KByte	
26.08.2009 00:00:00 - 00:00:00	9.059.378 KByte	859 kbit/s	2.333.226 KByte	
25.08.2009 00:00:00 - 00:00:00	8.400.623 KByte	797 kbit/s	1.656.687 KByte	



Loading...

V 7.2.0.4415

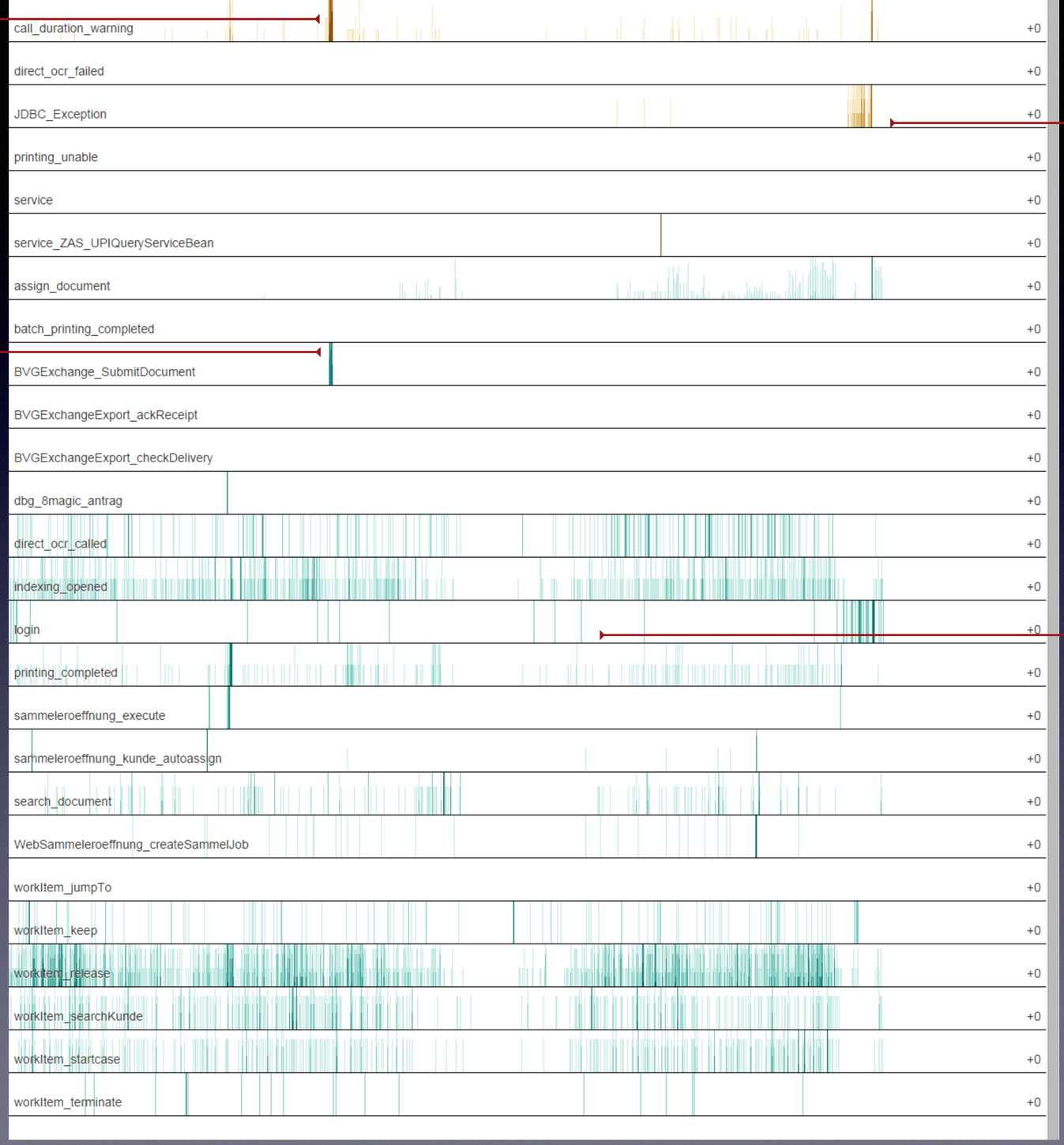
call duration

JDBC Exception

WS call

login

timeseries



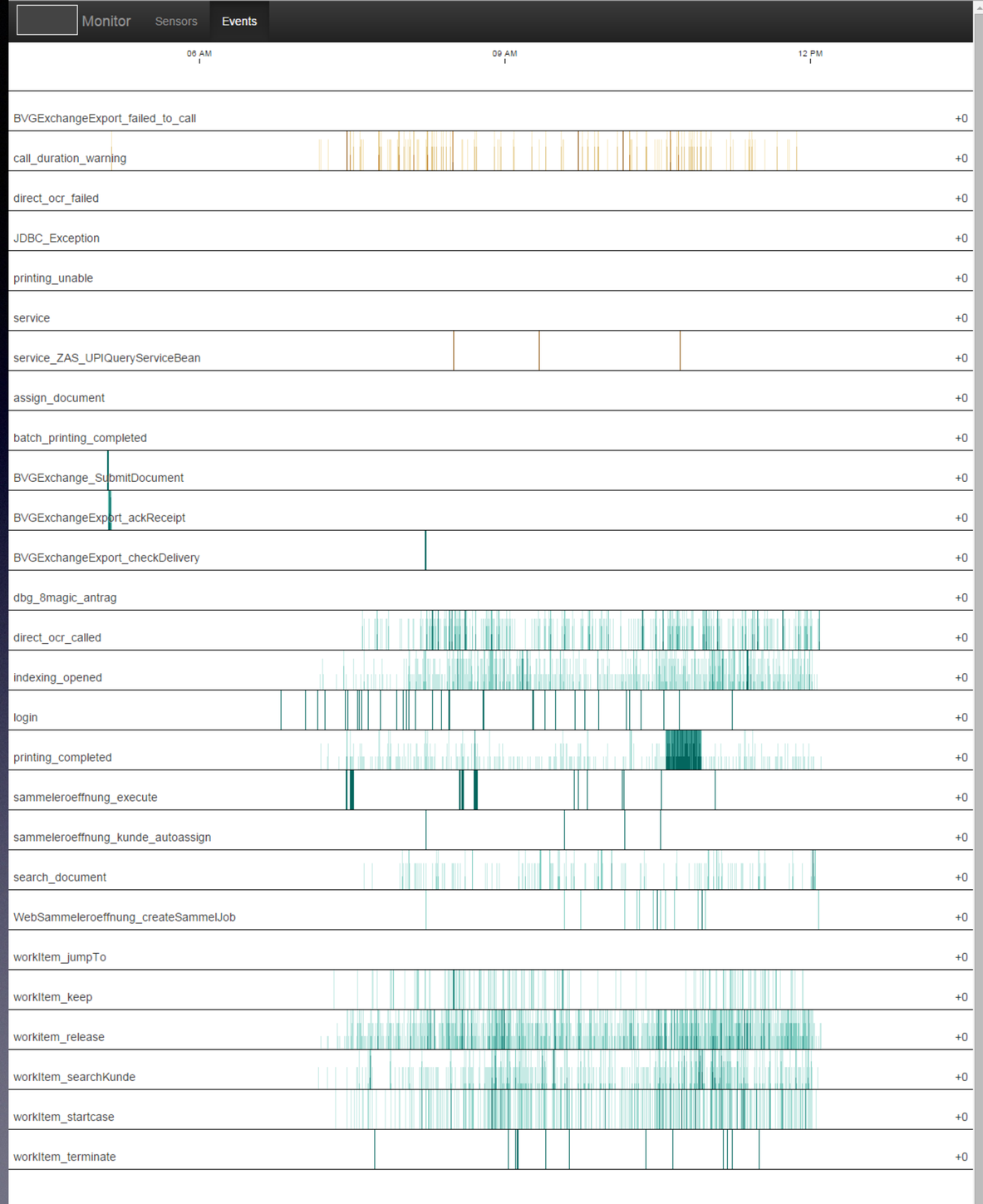
```
93      go collectAndSave()
94      for {
95          t, err := tail.TailFile(ac.EventTriggerDefaults.File, tail.Config{Poll: true, Follow: true})
96          if err != nil {
97              panic(err)
98          }
99
100         for line := range t.Lines {
101
102             for _, et := range ac.EventTriggers {
103                 e := &NoEvent
104                 if et.Category == "" {
105                     break
106                 }
107                 switch {
108                 case et.MatchBy == "contains":
109                     e = parseContains(et, line.Text)
110                 case et.MatchBy == "regexp":
111                     e = parseRegexp(et, line.Text)
112                 }
113
114                 if e != &NoEvent && e.Time.After(toStart) {
115                     e.Target = ac.EventTriggerDefaults.Target
116                     e.Sender = ac.EventServer.Sender
117                     go createAndSave(e)
118                     break
119                 }
120             }
121         }
122     }
123 }
```

```
var SAVE_SIZE = 256
var TIMEOUT = time.Duration(20)
var saveChannel = make(chan *MonitorEvent, 1)

func collectAndSave() {

    toSave := []*MonitorEvent{}
    for {
        select {
        case <-time.After(TIMEOUT * time.Second):
            for {
                select {
                // prevent timeout
                case me := <-saveChannel:
                    toSave = append(toSave, me)
                    if len(toSave) >= SAVE_SIZE {
                        saveEvents(toSave)
                        toSave = []*MonitorEvent{}
                    }
                    break
                // prevent thunders
                case <-time.After(100 * time.Millisecond):
                    if len(toSave) > 0 {
                        saveEvents(toSave)
                        toSave = []*MonitorEvent{}
                    }
                    break
                }
            }
        }
    }
}
```

first try

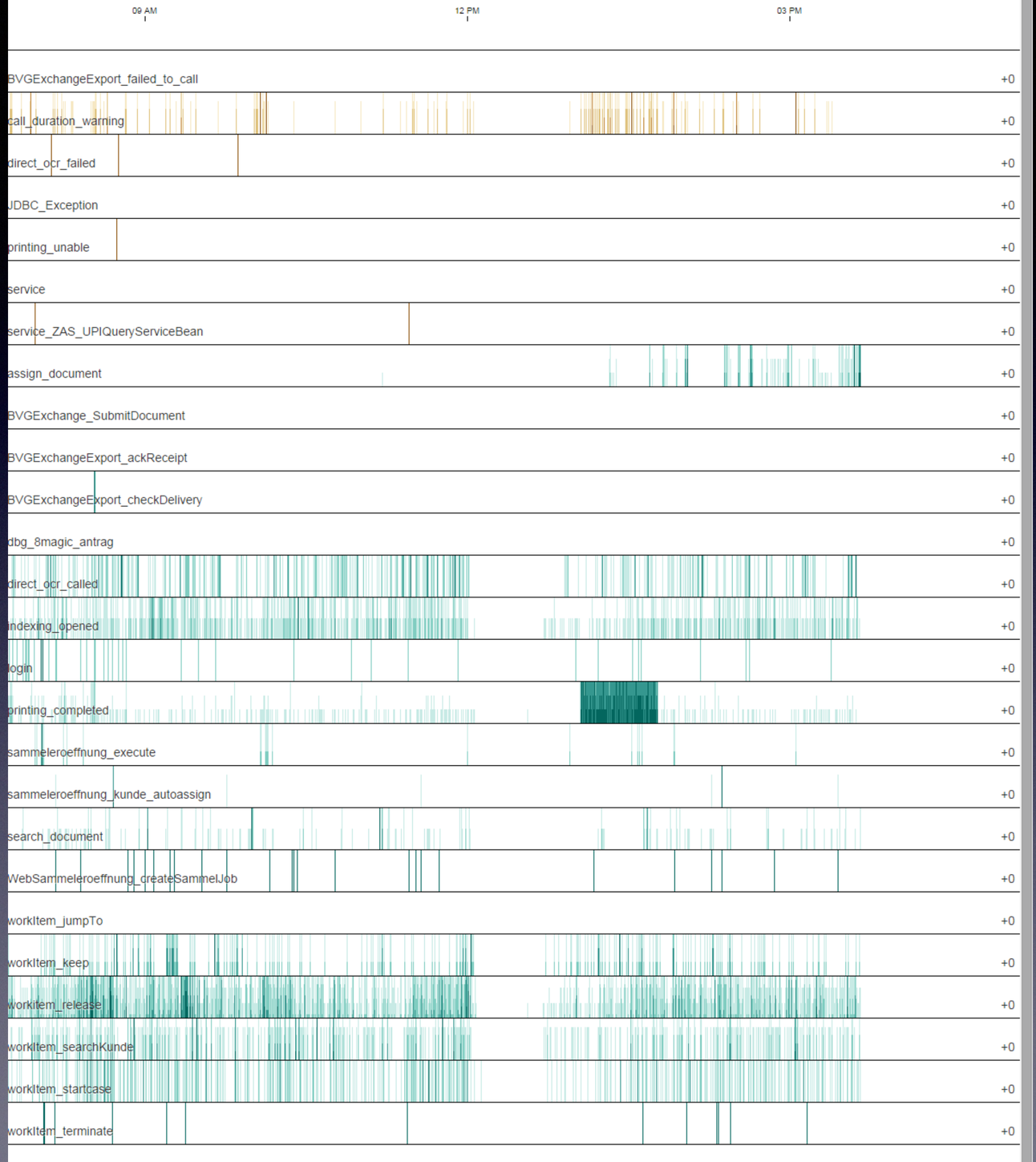

localhost:8080/Events/cubism/
03 AM
06 AM
09 AM
12 PM
03 PM
06 PM
360e8edf50bf07d6c9beeac6b3222c2157c3f9f7
+0.0%
28e483146a00d19e854d30a210fcf2d35e9ab3bb
+0.0%
0d430284a26a7dce981d04fc83185ae7c4f96925
+0.0%
b0a80eea686f39e4d5608783e744dd5418e5d7f1
+0.0%
7654cc94f814e02577711f23938ef42bc90bd38c
+0.0%
03a55830eb3d00f22b78f88490fface56dd88fab
+0.0%
81b763efe7815fb462238c25e646aa07d1cefe9f
+0.0%
4be609e14a8821c58b01c634d205ed6a03f2eb40
+0.0%
2c34f335538453105978fefd605e954069664945
+0.0%

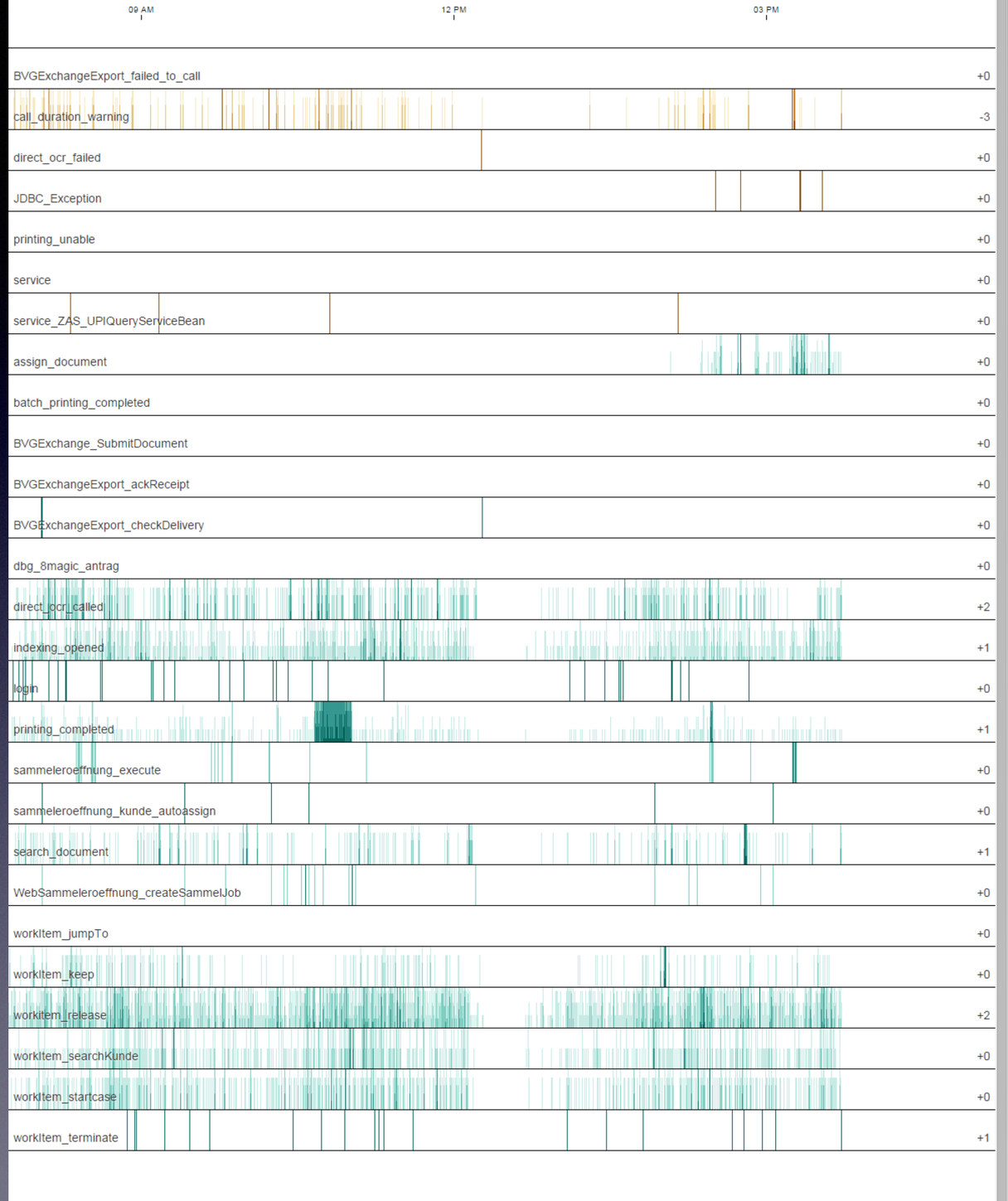
a morning


Monitor
Sensors
Events
06 AM
09 AM
12 PM
BVGExchangeExport_failed_to_call +0
call_duration_warning +0
direct_ocr_failed +0
JDBC_Exception +0
printing_unable +0
service +0
service_ZAS_UPIQueryServiceBean +0
assign_document +0
batch_printing_completed +0
BVGExchange_SubmitDocument +0
BVGExchangeExport_ackReceipt +0
BVGExchangeExport_checkDelivery +0
dbg_8magic_antrag +0
direct_ocr_called +0
indexing_opened +0
login +0
printing_completed +0
sammeleroeffnung_execute +0
sammeleroeffnung_kunde_autoassign +0
search_document +0
WebSammeleroeffnung_createSammelJob +0
workItem_jumpTo +0
workItem_keep +0
workitem_release +0
workItem_searchKunde +0
workItem_startcase +0
workItem_terminate +0

correlate events

working day 1



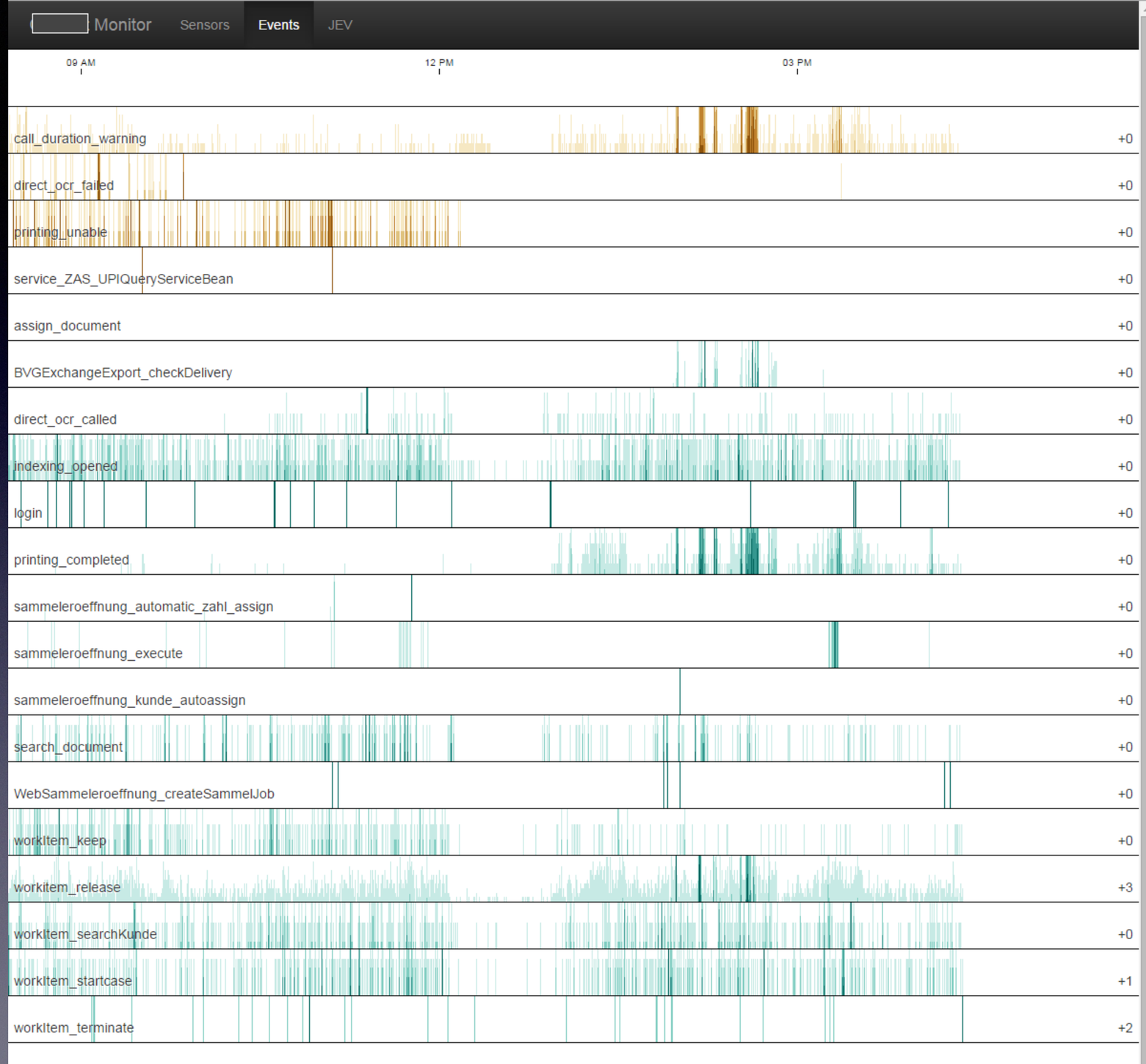


working day 2

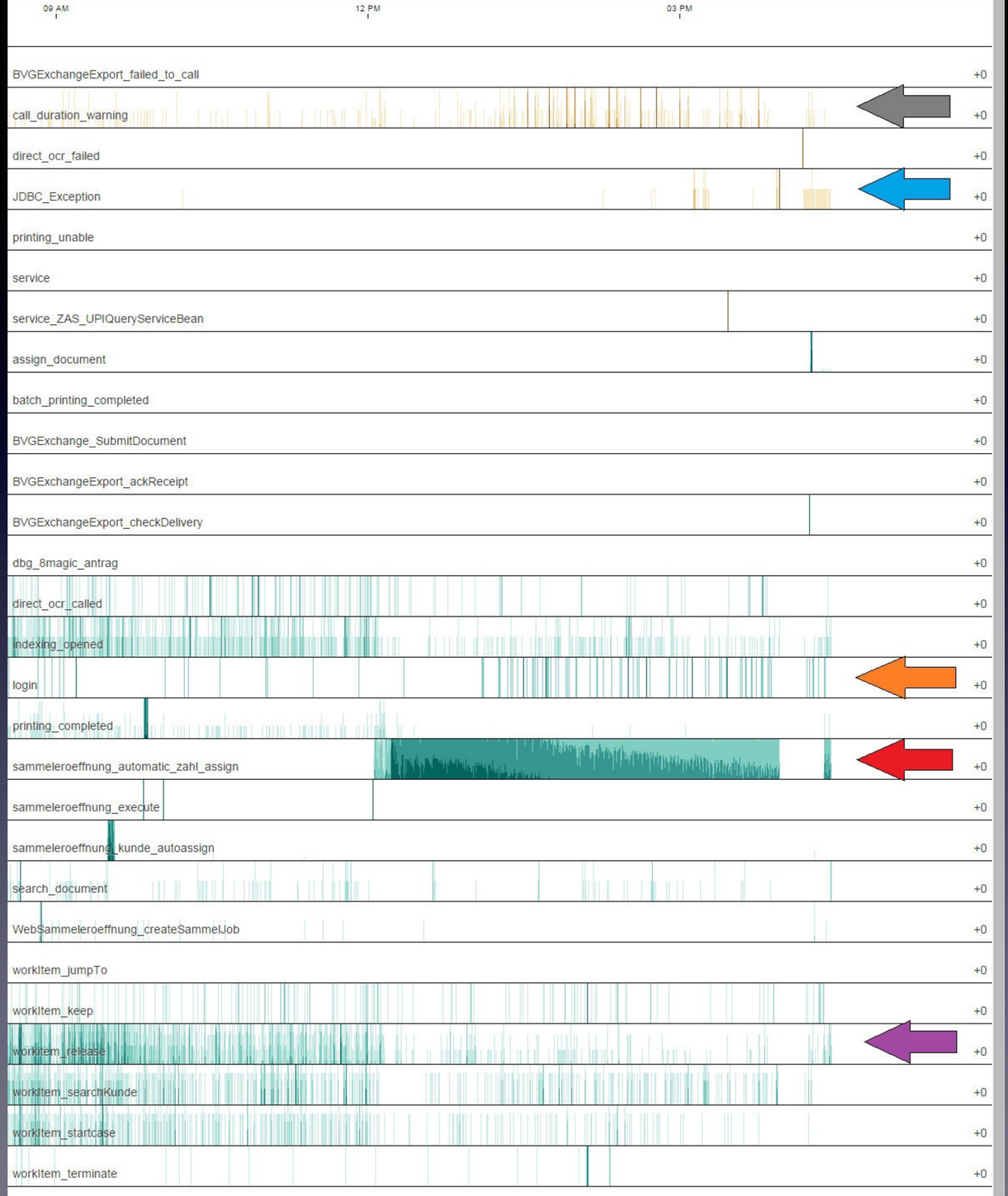
what is this?

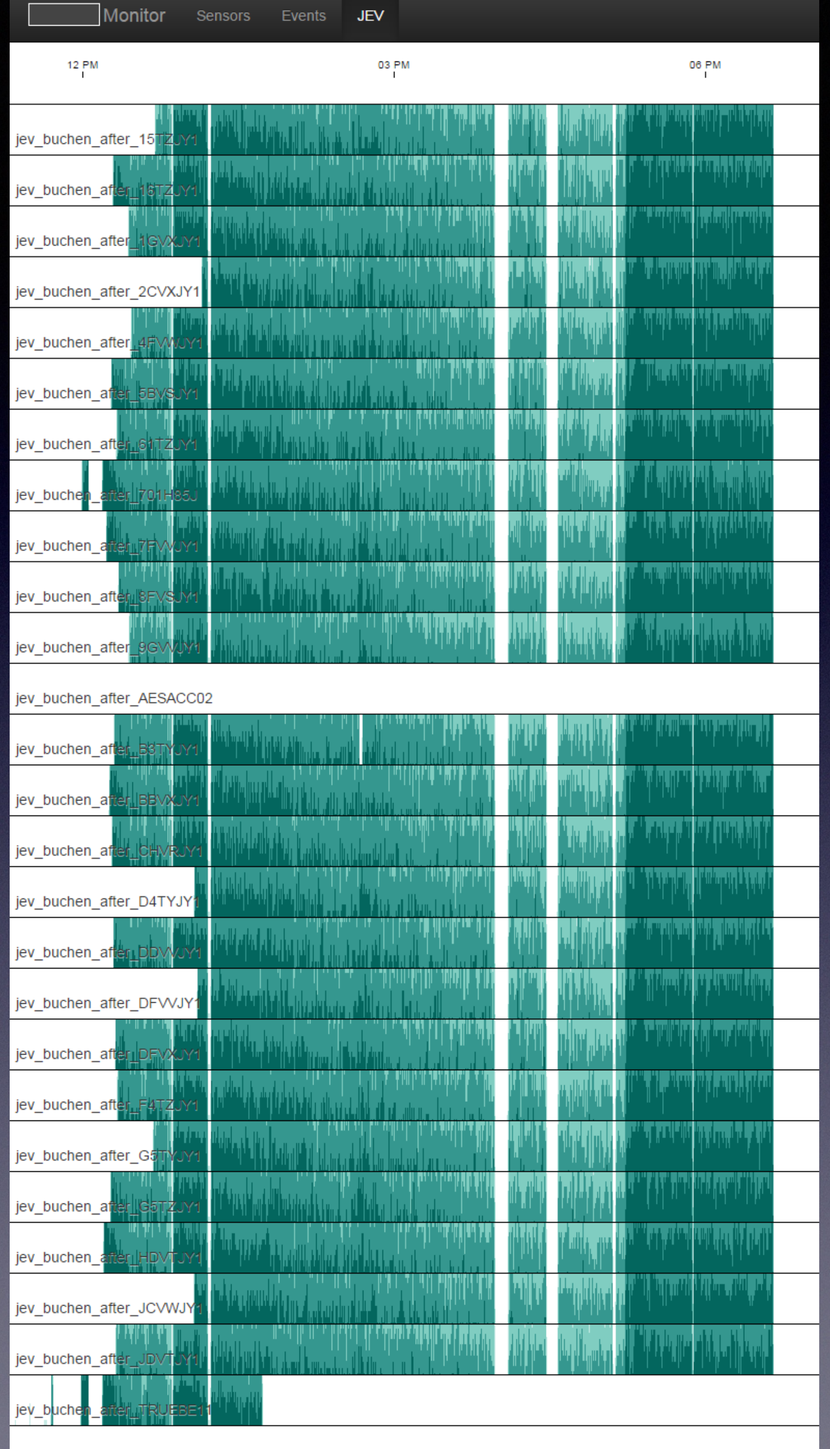

09 AM
12 PM
03 PM
call_duration_warning
direct_ocr_failed
JDBC_Exception
printing_unable
service_ZAS_UPIQueryServiceBean
assign_document
BVGExchange_SubmitDocument
BVGExchangeExport_ackReceipt
BVGExchangeExport_checkDelivery
dbg_8magic_antrag
direct_ocr_called
indexing_opened
login
printing_completed
sammeleroeffnung_automatic_zahl_assign
sammeleroeffnung_execute
sammeleroeffnung_kunde_autoassign
search_document
WebSammeleroeffnung_createSammelJob
workItem_jumpTo
workItem_keep
workitem_release
workItem_searchKunde
workItem_startcase
workItem_terminate
+0

printing Issues



Post Mortem


09 AM
12 PM
03 PM
BVGExchangeExport_failed_to_call +0
call_duration_warning +0
direct_ocr_failed +0
JDBC_Exception +0
printing_unable +0
service +0
service_ZAS_UPIQueryServiceBean +0
assign_document +0
batch_printing_completed +0
BVGExchange_SubmitDocument +0
BVGExchangeExport_ackReceipt +0
BVGExchangeExport_checkDelivery +0
dbg_8magic_antrag +0
direct_ocr_called +0
indexing_opened +0
login +0
printing_completed +0
sammeleroeffnung_automatic_zahl_assign +0
sammeleroeffnung_execute +0
sammeleroeffnung_kunde_autoassign +0
search_document +0
WebSammeleroeffnung_createSammelJob +0
workItem_jumpTo +0
workItem_keep +0
workItem_release +0
workItem_searchKunde +0
workItem_startcase +0
workItem_terminate +0

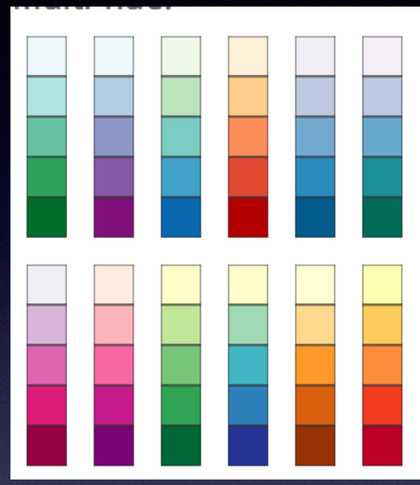


JEV
(Jahresendverarbeitung)

- 24 “Node Cluster” JBoss
- 1 Batch Master
- year before took x hours

=> distribute JBoss + config with “ClientStarter”
=> monitor it with EventAgent

visual perception



- colors (Cyntia Brewer)
- correlate events
- event count is not that important (one, a few, a lot)
- see: http://edwardtufte.com/ - http://colorbrewer2.org
- “PowerPoint Does Rocket Science”

using Go? (1)

- Senior JAVA Developer: ”I can do that with JAVA too"
- how to deploy that in a JVM based ops infrastructure?
- which IDE do you use for that?

using Go? (2)

- stable and reliable
- nice programming environment, also for Windows
- easy to learn, docs and spec are enough
- POLA, (principle of least astonishment)

Why Go is fun ?

- Go is boring (kind of)… but thats good
  - simple, elegant, orthogonal, POLA
- interfaces (not the same as JAVA, might be confusing for them)
- channels
- select

Questions ?..
Thank you very much

2015-04-01

Marcel Lanz
Senior System & Software Architect
marcel.lanz@procentric.ch