

Splunk at UniCredit

Our Big Data Journey from Daily Troubleshooting to Business Analytics

Marcello Bianchetti, ICT Solution Feasibility - UniCredit

Strata + Hadoop World Barcelona 2014 November 20th , 2014

Agenda

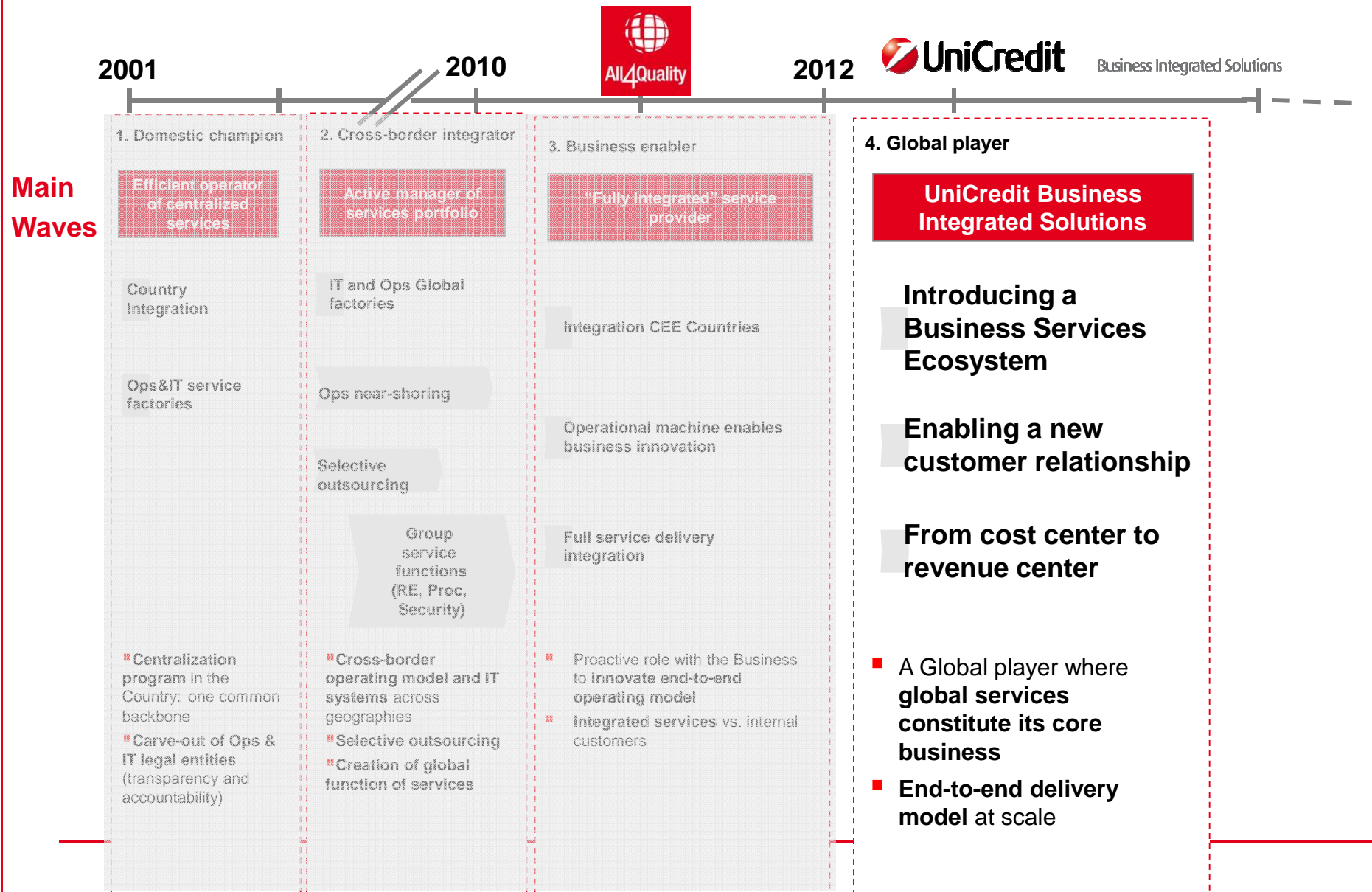
- About us
- Why Splunk
- Orientation: Splunk basic concepts
- Splunk architecture at UniCredit
- Splunk implementation
- Technical achievements & lessons learnt
- Centralized automated agent update
- Centralized cronjob scheduling
- Advanced system monitoring
- OS Tuning
- Use cases

About us: UniCredit at a glance

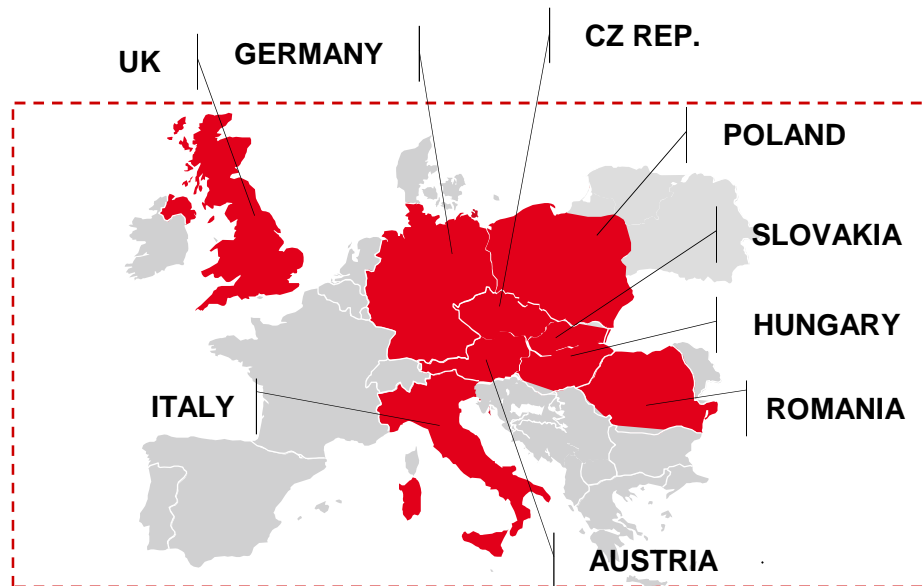


* Source: UniCredit Company Profile, data as at June 30, 2014

About us: From Instrumental Services to Business Integrated Solutions



About us: UniCredit Business Integrated Solutions at a glance



- ~ 10.000 Fte's*
(y/y pro forma)
- 11 Countries**
- 4 Wholly-owned subsidiaries*

- NET EQUITY: € 406,011,164*
- TOTAL REVENUE: € 2,328,665,282*

UniCredit Business Integrated Solutions is the first concrete milestone within the Group Strategic Plan 2012 announced in November 2011 to be achieved.

Owned by UniCredit, is created from the integration and consolidation of 16 Group companies (among them UGIS, UCBP, URE, UC) and is dedicated to providing services in the sectors of Information and Communication Technology (ICT), Back Office and Middle Office, Real Estate, Security and Procurement.

A new business model, unique in the European banking sector, **focused on Business needs** (i.e. Commercial Banking, Global Markets, CEE), not only on providing services.

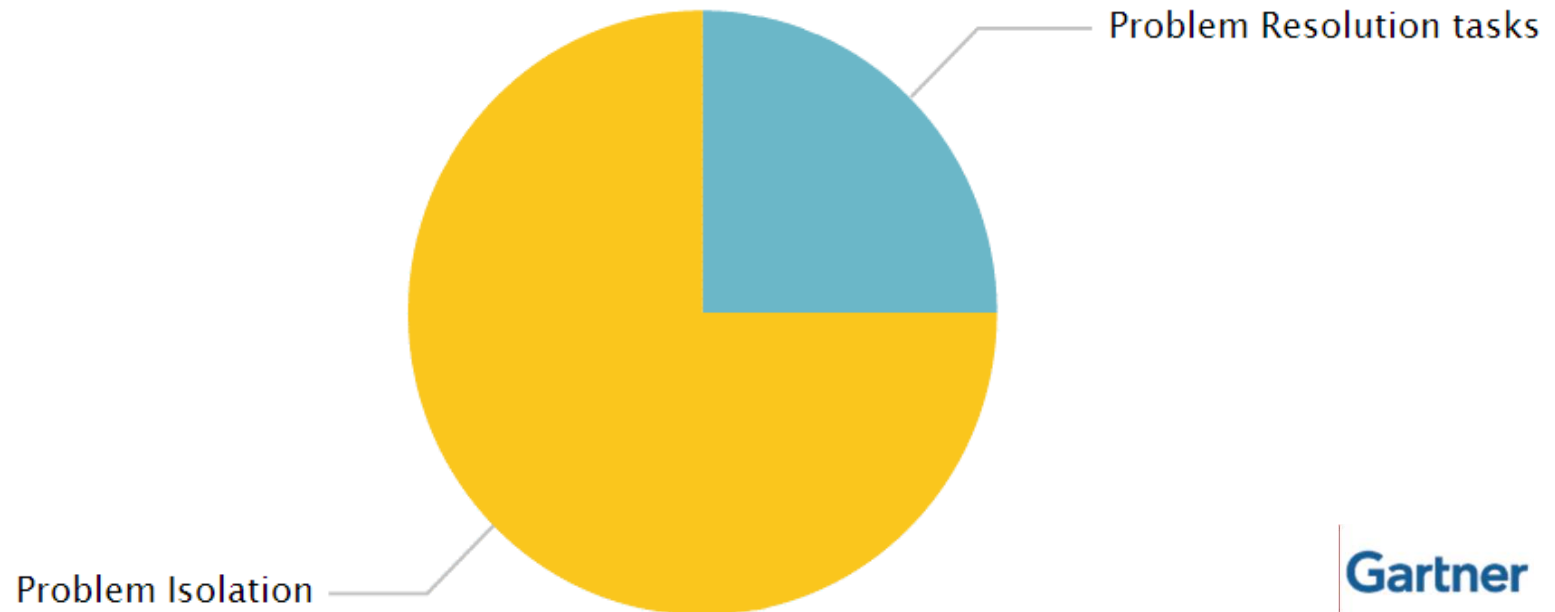
* Data as at December 31, 2013.

** UniCredit Business Integrated Solutions operates also in 2 branches, one located in New York and one in Singapore.

Why Splunk: Troubleshooting complex problems

- Highly distributed and multilayered SOA architectures
- Huge number of log files
- Logs contain 70% of diagnostic data useful for problem isolation

Problem Resolution time



Gartner

Why Splunk: A flexible solution to a complex problem

Needs / Activities 								
	Res Consumption	Workload	Gather Logs	Describe Data	Cartography	KPI Thresholds	Secure Access	Interactive UI
Visibility (IT Systems)								
Visibility (IT Services)								
Faster Employees Onboarding								
Effective Incident Management								
Simplified Collaboration								
Allow externals to access data								
Define Alerting								

Orientation: Splunk basic concepts

■ Processing at the time the data is processed

- Splunk reads data from a **source**, such as a file or port, on a host (e.g. "my machine"), classifies that source into a **sourcetype** (e.g., "log4j", "messages", "access_combined", ...), then extracts timestamps, breaks up the source into individual events (e.g., log events, alerts, ...), which can be a single-line or multiple lines, and writes each event into an **index** on disk, for later retrieval with a search.

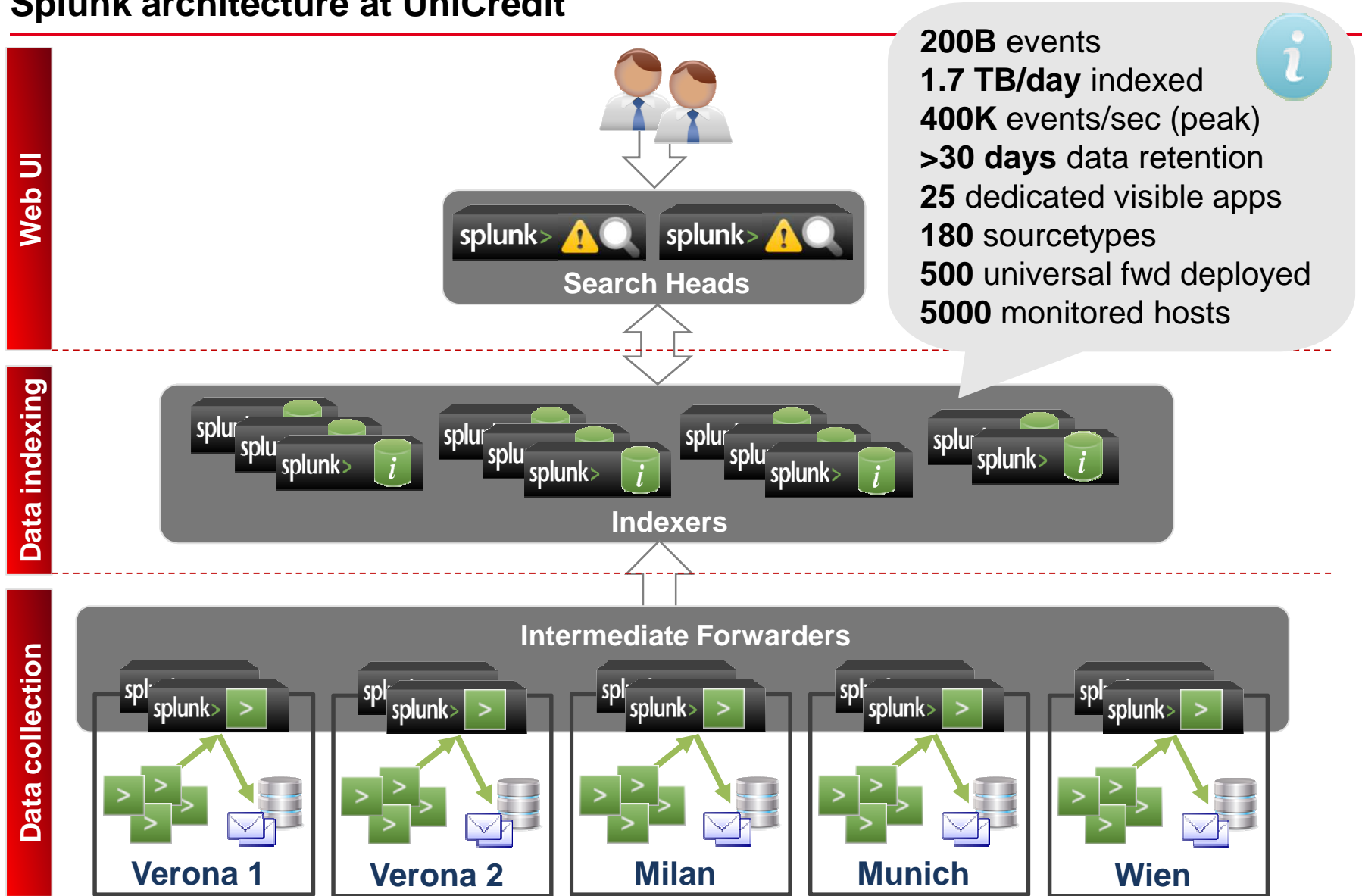
■ Processing at the time the data is searched

- When a search starts, matching indexed events are retrieved from disk, **fields** (e.g., user=Mark, product=Mobile, ...) are extracted from the event's text, and the event is classified by matching it against **eventtype** definitions (e.g., 'error', 'login', ...).

■ Fields

- Fields are searchable name/value pairings in event data. As Splunk processes events at index time and search time, it automatically extracts fields. At index time, Splunk extracts a small set of default fields for each event, including host, source, and sourcetype. At search time, Splunk extracts what can be a wide range of fields from the event data, including user-defined patterns (e.g., the user and sessionid fields) as well as obvious field name/value pairs such as product=Mobile.

Splunk architecture at UniCredit



Splunk implementation: Project timeline

Release 1

3 Indexers
Up to 250 GB/day

February 2011

- **Data:** ~ application server logs
- **Users:** ~ Service Desk
- **Use case:** incident management, monitoring, troubleshooting
- **Splunk role:** tool for the ServiceDesk

Release 2

8 Indexers
Up to 700 GB/day

January 2013

- **Data:** +webservers, +DB2, +system metrics, ...
- **Users:** +Developers, +Control Room, +Consultants, +Executives, +SysAdmins
- **Use case:** +application management, +alarms, +IT & business reporting
- **Splunk role:** Enterprise standard

Latest Release

11 Indexers
Up to 1.7 TB/day

October 2014

- **Data:** +mainframe data, +network appliances, ...
- **Use case:** +long term capacity planning, +fraud analysis
- **Splunk role:** Enterprise standard

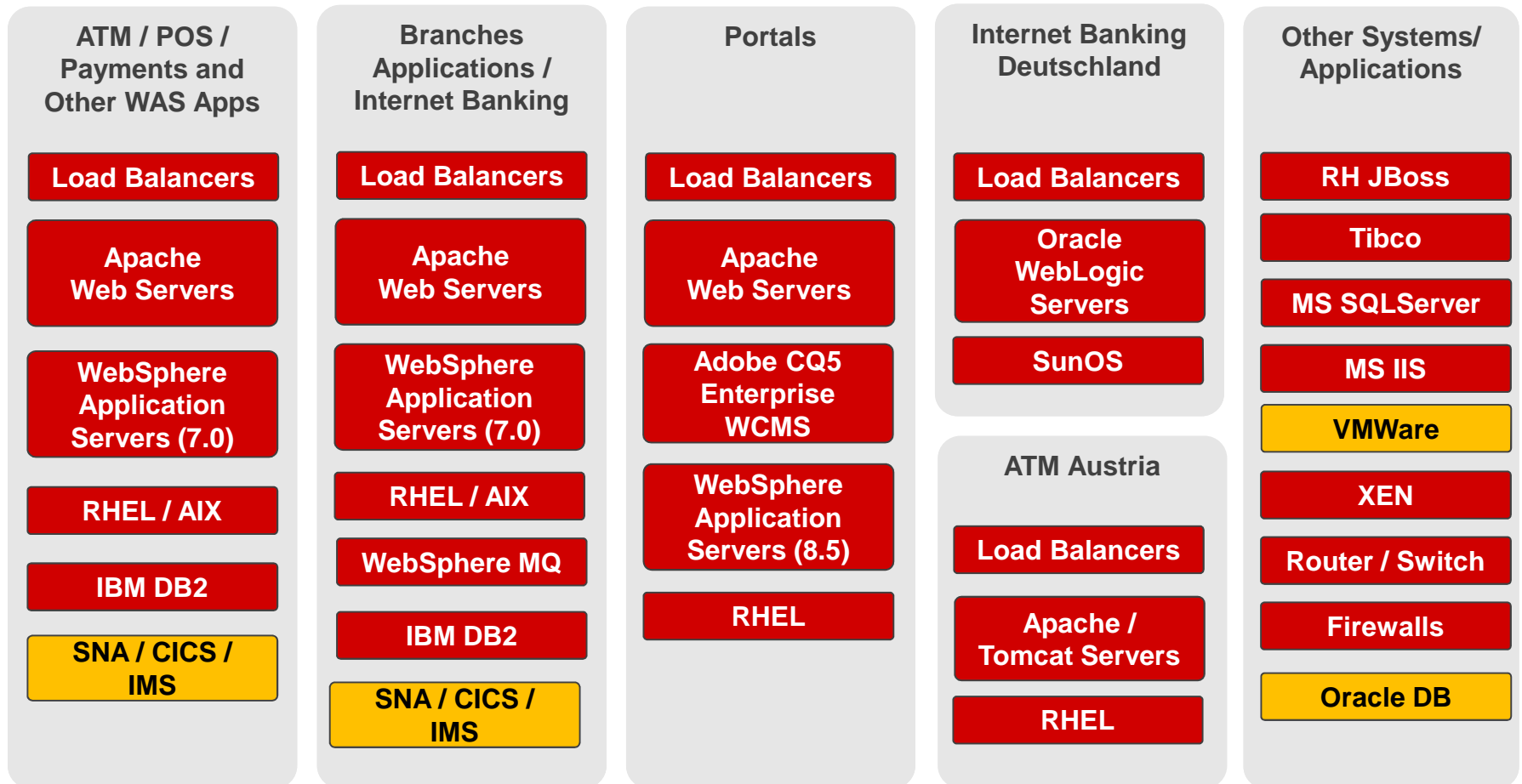
Next Releases

Up to 6,3 TB/day

... 2015

- **Data:** +auditing data, +security logs, +configuration data, +service KPIs, +Hadoop data, ...
- **Use case:** +compliance, +security, +configuration discovery, +SLA monitoring, +Hadoop integration, +docker containers monitoring
- **Splunk role:** Enterprise SIEM & other small dedicated instances

Splunk implementation: Scope



current

In progress

Technical achievements & lessons learnt

Our Pride

Advanced system monitoring

- Detailed resource accounting
- Network connections by process

Centralized cronjob scheduling

- Execute data collection scripts even when the agent is down.

Centralized automated agent update

- Easy roll-out of new Splunk version

Lesson Learnt

Consider Parsing phase

- Check the size of events (big xml, loops, etc)
- Limit linebreaking errors to achieve better performance

Define conventions

- Index strategy
- Sourcetype naming strategy
- Saved search and dashboard naming conventions

Centralized config. Deployment

- Beware of the restarts!
- Independent configurations, apps
- Multilayered deployment (use VLAN, avoid Firewall influence)

OS Tuning

- Network tuning
- Disable transparent huge pages

Centralized automated agent update

- For each platform an app has been created containing a very simple inputs.conf file, the binary package, two shell scripts and a text file:
 - splunkforwarder-release.txt
 - splunkufupgrade_check.sh
 - splunkufupgrade.sh
 - splunkforwarder-Linux-x86_64.tgz

./default/inputs.conf

```
[script://./bin/splunkufupgrade_check.sh 2>/dev/null 1>/dev/null]  
disabled = false  
interval = 86400
```

./bin/splunkforwarder-release.txt

5.0.6

Centralized automated agent update

`./bin/splunkufupgrade_check.sh`

```
#!/bin/bash

ARCH=$(uname)
SCRIPTDIR=$SPLUNK_HOME/etc/apps/u_upgrade_${ARCH}/bin

NEWVER=$(cat $SCRIPTDIR/splunkforwarder-release.txt)
CURVER=$(($SPLUNK_HOME/bin/splunk version|awk '{ print $4 }')

if [ "$NEWVER" \> "$CURVER" ]; then
    . $SCRIPTDIR/splunkufupgrade.sh &
fi
```

Centralized automated agent update

./bin/splunkufupgrade.sh

```
#!/bin/bash

$SPLUNK_HOME/bin/splunk stop
sleep 2
chmod -R +w $SPLUNK_HOME/
if [ "$ARCH" = "AIX" ]; then
    sudo /usr/sbin/slibclean
fi
cd $SPLUNK_HOME/..
gunzip -c $SCRIPTDIR/$INSTARCHIVE | tar xf -
if [ "$ARCH" != "solaris" ]; then
    ulimit -m unlimited 2>&1
else
    ulimit -v unlimited 2>&1
fi
$SPLUNK_HOME/bin/splunk start --answer-yes --no-prompt --accept-license
```

Centralized cronjob scheduling

- For all *nix platforms an app has been developed in order to use system cron facility for scheduling bash/perl/awk scripts:
 - `cronedit.sh`
 - edit a single line of the splunk user crontab
 - `cronmanager.sh`
 - check the `cron_config.conf` file and call the `cronedit.sh` for each line to be scheduled/unscheduled
 - `cron_config.conf`
 - `cron_inputs_{os_name}.txt`

`./default/inputs.conf`

```
[script://./bin/cronmanager.sh configs/cron_config.conf]
disabled = false
interval = 3600
index=sys
sourcetype=adm:cronmanager
```

Centralized cronjob scheduling

`./bin/configs/cron_config.conf`

```
Linux    usspupl* configs/cron_inputs_splunk.txt
Linux    *      configs/cron_inputs_linux.txt
AIX      *      configs/cron_inputs_aix.txt
SunOS    *      configs/cron_inputs_solaris.txt
```

`./bin/configs/cron_inputs_linux.txt`

```
# Please insert one line for every crontab add/disable/remove operation.
# The syntax should be:
# <on|off|remove> "<crontab_schedule>" $SPLUNK_HOME/<app_bin_folder>/
#                                     <script_name>.sh [parameters]*

on "9 * * * *" $SPLUNK_HOME/etc/apps/u_acct_inputs/bin/df_acct_log.sh 2>
                                     $SPLUNK_HOME/var/log/crond/df.err
on "* * * * *" $SPLUNK_HOME/etc/apps/u_acct_inputs/bin/ps_acct_log.sh 2>
                                     $SPLUNK_HOME/var/log/crond/ps.err
on "* * * * *" $SPLUNK_HOME/etc/apps/u_acct_inputs/bin/lsof_acct_log_v2.sh 2>
                                     $SPLUNK_HOME/var/log/crond/lsof.err
on "* * * * *" $SPLUNK_HOME/etc/apps/u_acct_inputs/bin/nmon_acct_log.sh 2>
                                     $SPLUNK_HOME/var/log/crond/nmon.err
```

Advanced system monitoring

- For all *nix platforms an app has been developed for collecting customized ps, nmon, df and lsof outputs

./bin/acct_lsof.sh

```
#!/bin/bash
ACCTLISOFPPL="$CURRDIR/acct_lsof_conn_dir.pl"
(...omissis...)
/usr/bin/sudo /usr/sbin/lsof -iTCP -P -b -l -n | grep 'java ' 2> /dev/null
    > $TMP_RAW_LSOF
egrep "\->" "$TMP_RAW_LSOF" | egrep -v "(\[|)|(:\*\->)" | sed "s/)//<
    | sed "s/IPv.*TCP//" | awk 'BEGIN {FS="->|:|\t| +";'"$PID_CLONE_ARRAY"' }
    {print "pid=" $2 " u_clone=" clone[$2] " loc_prt=" $6 " rem_ip=" $7 "
    rem_prt=" $8 " status=" substr($9,2) }' > $TMP_SPLUNK_LSOF
# LISTEN ports
grep LISTEN "$TMP_RAW_LSOF" | sed "s/)//<" | sed "s/IPv.*TCP//" | awk 'BEGIN
    {FS="->|:|\t| +";'"$PID_CLONE_ARRAY"' } { print "pid=" $2 " u_clone="
    clone[$2] " loc_prt=" $6 " status=" substr($7,2) }' | grep -v "loc_prt= <
    | tee $TMP_LOG | awk '{print $3}' | awk -F= '{print $2}' > $LST_PORTS_FILE
# ESTABLISHED Connections
grep ESTABLISHED "$TMP_SPLUNK_LSOF" | tee $TMP_EST_CONN | awk '{print $3}'
    | awk -F= '{print $2}' > $EST_PORTS_FILE
# Other status connections
grep -v ESTABLISHED "$TMP_SPLUNK_LSOF" | grep -v LISTEN >> $TMP_LOG
$ACCTLISOFPPL $LST_PORTS_FILE $EST_PORTS_FILE | paste -d " " "$TMP_EST_CONN -
    >> $TMP_LOG
awk '{print "'$DATE'" " $0}' $TMP_LOG >> $LOG
```

Advanced system monitoring

./bin/acct_ps.pl

```
#!/usr/bin/perl -w
our $PS_CMD_JAVA = "ps -eo pid,thcount,cputime,vsz,args|grep 'java '
    |grep -v grep|awk '{ print \$5 \"@\" \"\$(NF-1) \"@\" \"\$NF \"#\" \"\$1 \" \"
    \$1 \" \" \"\$2 \" \" \"\$3 \" \" \"\$4 }'|sort";
our $PS_CMD_JAVA_SUNOS = "ps -eo pid,nlwp,time,vsz,args|grep 'java '
    |grep -v grep|awk '{ print \$5 \"@\" \"\$(NF-1) \"@\" \"\$NF \"#\" \"\$1 \" \"
    \$1 \" \" \"\$2 \" \" \"\$3 \" \" \"\$4 }'|sort";
our $SLZ_PATH = '/var/tmp/splunk/splunk_slz_ps.slz';
(...omissis...)
$ps_data = get_ps_data();
$ps_last_data = load($SLZ_PATH);
store($ps_data, $SLZ_PATH);
foreach $clone (keys %$ps_data) {
    if (exists $$ps_last_data{$clone}) {
        $time_delta = $$ps_data{$clone}{'ts'} - $$ps_last_data{$clone}{'ts'};
        if ($$ps_data{$clone}{'pid'} == $$ps_last_data{$clone}{'pid'}) {
            $cputime_delta = $$ps_data{$clone}{'cputime'} -
                $$ps_last_data{$clone}{'cputime'};
        } else {
            $cputime_delta = $$ps_data{$clone}{'cputime'};
        }
        $dcpu_dt = $cputime_delta / $time_delta;
        (...omissis...)
        print "[\$time] pid=$pid threads=$th_count vsz_kb=$vsz
            dcpu=$cputime_delta dcpu_dt=$dcpu_dt$u_clone process=$process_id\n";
    }
}
```

OS Tuning

- Network tuning on Intermediate Forwarders as well as on Indexers:

/etc/sysctl.conf

```
(...omissis...)
net.ipv4.tcp_synack_retries = 3
net.ipv4.tcp_syn_retries = 4
net.ipv4.tcp_fin_timeout = 30
net.ipv4.tcp_orphan_retries = 4
net.ipv4.tcp_keepalive_intvl = 15
net.ipv4.tcp_keepalive_probes = 5
net.ipv4.tcp_keepalive_time = 300
net.core.rmem_default = 256960
net.core.wmem_default = 256960
net.core.rmem_max = 4194304
net.core.wmem_max = 4194304
net.ipv4.tcp_mem = 256960 256960 4194304
net.ipv4.tcp_rmem = 256960 256960 4194304
net.ipv4.tcp_wmem = 256960 256960 4194304
net.core.netdev_max_backlog = 3000
```

OS Tuning

- Disable transparent Huge Pages (because dirty pages deallocation is really bad for Splunk performance):

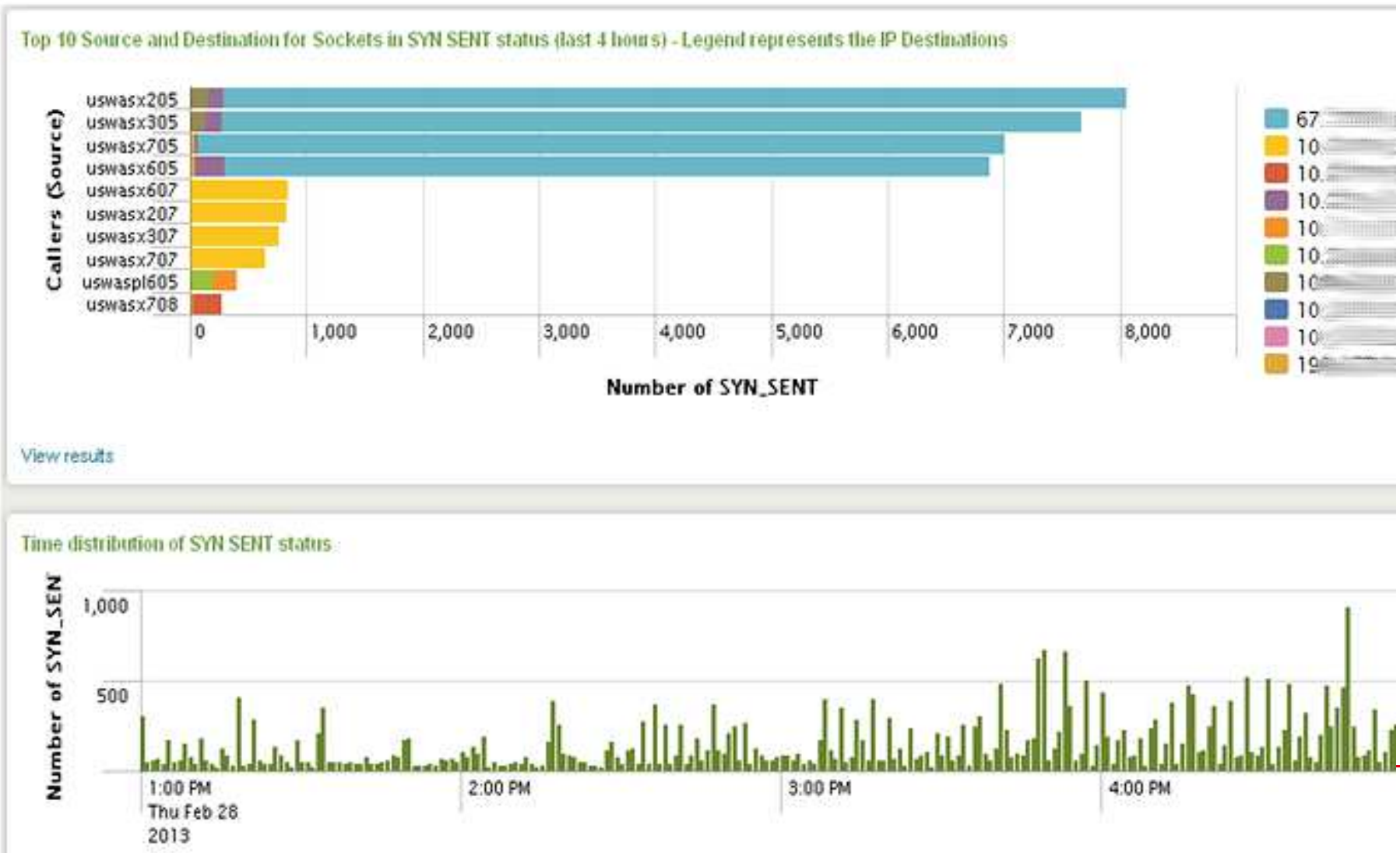
/etc/grub.conf

```
(...omissis...)
title Red Hat Enterprise Linux (2.6.32-279.el6.x86_64)
root (hd0,0)
kernel /vmlinuz-2.6.32-279.el6.x86_64 ro root=/dev/mapper/myvg-rootlv
\ rd_NO_LUKS rd_LVM_LV=myvg/rootlv LANG=en_US.UTF-8 KEYBOARDTYPE=pc
\ KEYTABLE=it rd_NO_MD crashkernel=auto crashkernel=auto rhgb quiet
\ SYSFONT=latarcyrheb-sun16 rd_NO_DM rhgb quiet transparent_hugepage=never
initrd /initramfs-2.6.32-279.el6.x86_64.img
```

```
$ echo never >/sys/kernel/mm/redhat_transparent_hugepage/enabled
```

Use cases – Network data analysis

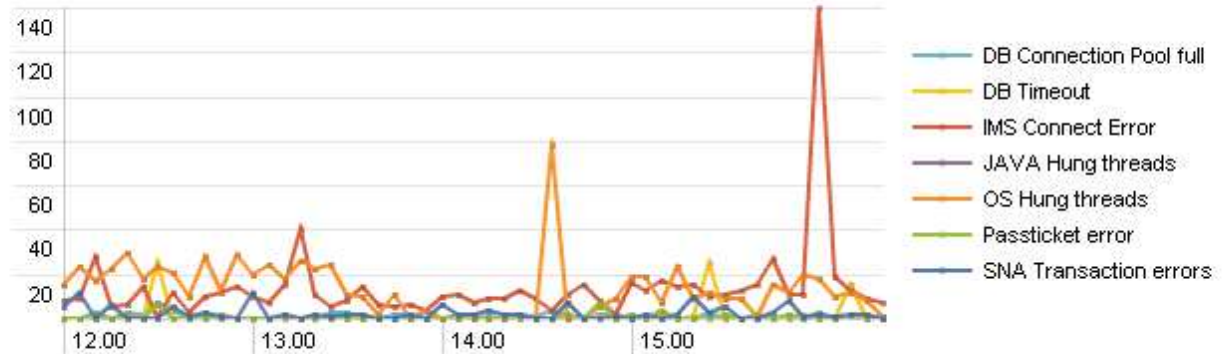
- Troubleshooting network problems for high distributed datacenters
- Analysis of systems and applications relationships



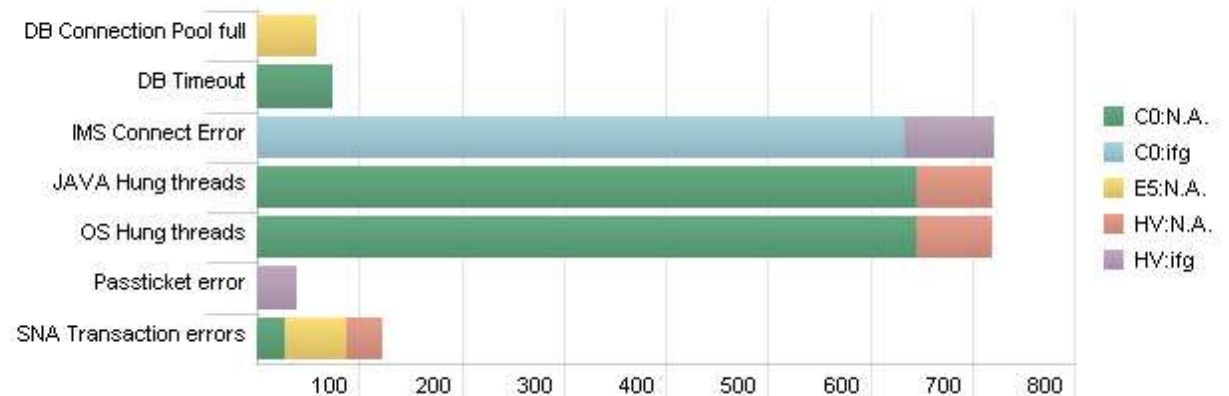
Use cases – Known error monitoring

- Historical or real-time analysis
- Immediate notification about common problems
- Spot trends and stop disruptive behavior before incidents happen

Errors over time



Errors by app and tower



Use cases – A foreign bank's internet banking

- **Application Management:** troubleshooting user problems
- **IT Operations:** checking the impact of incidents

IBC: Active users (with activity in the last 10 min) real-time



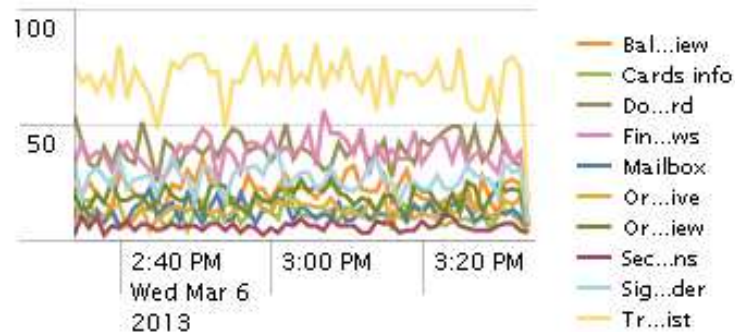
[View results](#)

IBC: Users connected real-time

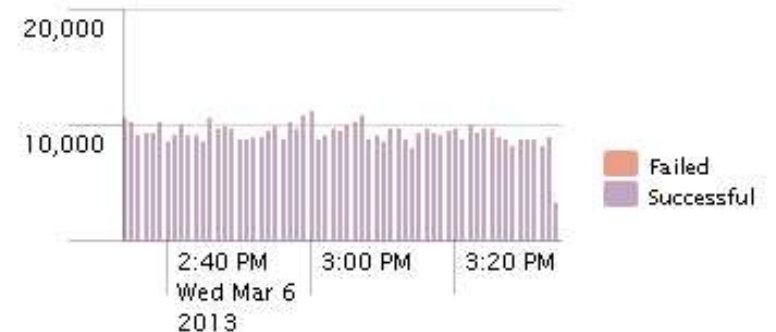


[View results](#)

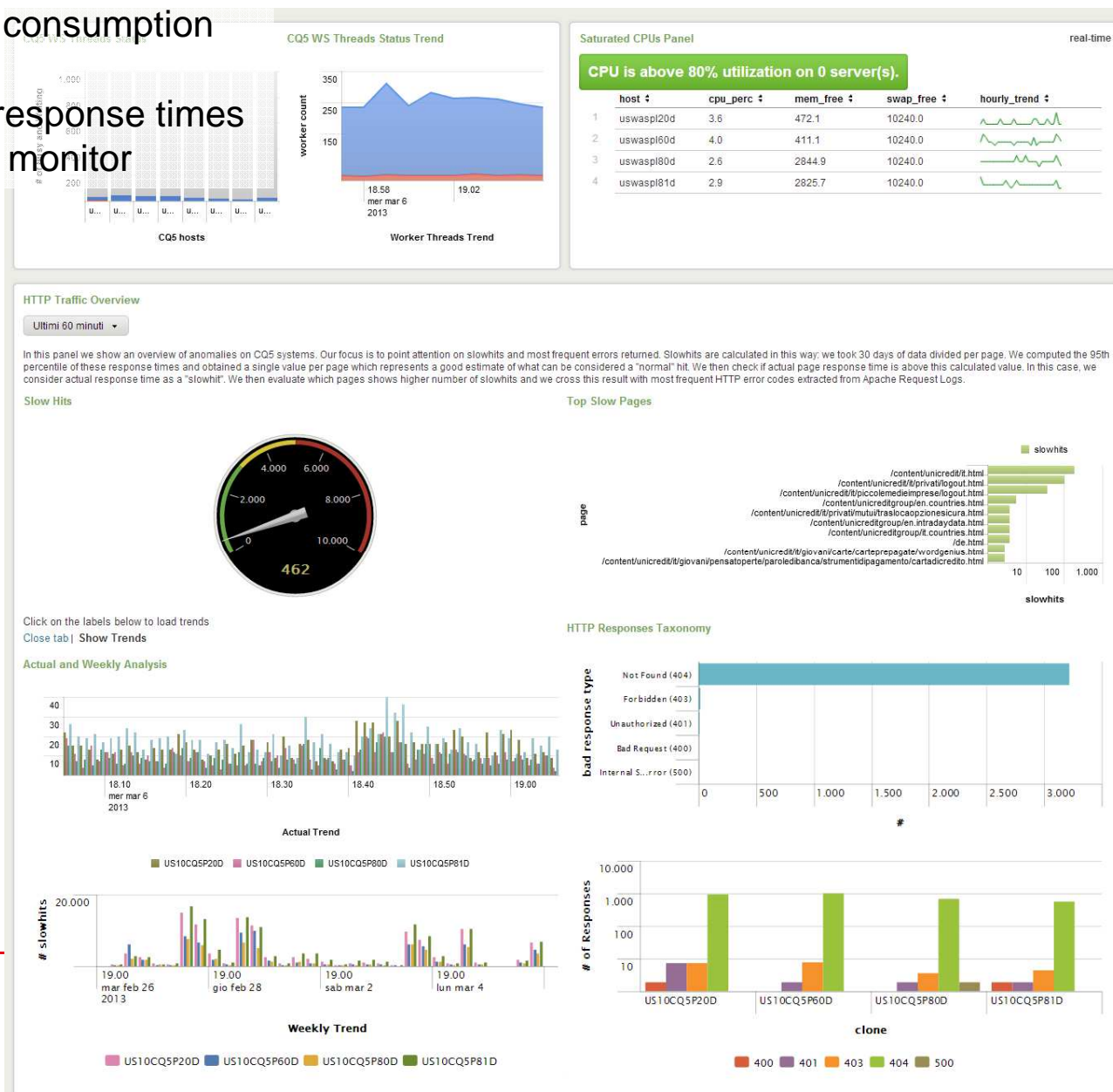
IBC: Top functions executed (top 10 values) 2m ago



IBP: Successful vs Failed Operations 1m ago



- From storage usage
- ... to system resource consumption
- ... to workload
- ... to page errors and response times
- A full-stack application monitor

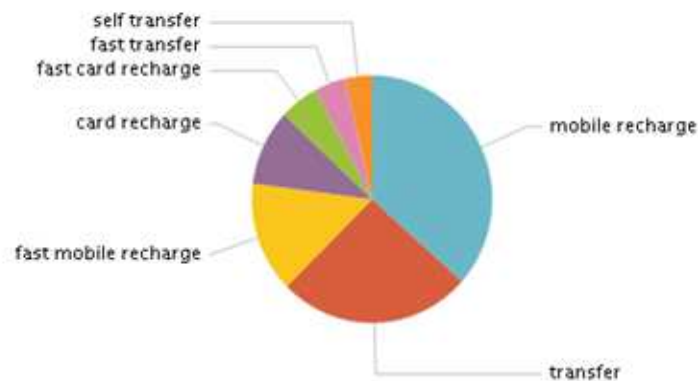


Use cases – UniCredit mobile banking

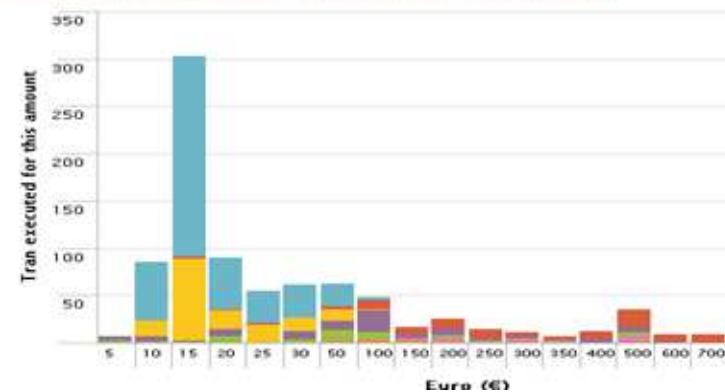
- **Application Management:** checking client application versions/upgrading procedures
- **Web Analytics:** monitoring client activity and navigation patterns
- **Business Analytics:** discovering common user behaviors and business trends



Money Moving Operations (Last 4 hours)



1s ago Number of dispositive transaction executed for each amount (Last 4h)



Use cases – Cards: business analytics based on POS and ATM transactions

- From an authorized card payment transaction record is possible to extract a lot of interesting business information:

Mastercard Circuit

```
[2014-09-04-14.45.54.608000] proc_source="B24A",
tmst_target="2013-09-04-14.45.54.724000", serv_id="ISS",
proc_input="MAST" proc_target="B24H", interface_acq="BNET_1",
interface_iss="02008", cod_msg="1110", oper_rrn="090448764439",
card_id="526430VS350Y2992", oper_amount="000000008000",
oper_currency="978", oper_country="380", term_id="00599307",
circuito="", sett_merc="4722", bin_acq="002111",
id_merc="329017246168" prcode="003000", action_code="000",
approval_code="H8H766" oper_mod_input="1", channel="0",
flag_dupl="Y", flag_onus="N", auth_rout_dst="INTFHI93",
auth_rout_id="HISO_AUTH", msg_subst="", ndq="0000032178507391",
station_acq="STA-BNET-MI1", acceptor="TRAWEL SPA\\MILANO\\380",
tmst_ins="2013-09-04-14.48.56.277466", lpar="B"
```

Merchant ID

Merchant sector

Merchant name

Client ID

Use cases – Cards: business analytics based on POS and ATM transactions

- Market share analysis
- Discover client behavior on card usage
- Regional business distribution

