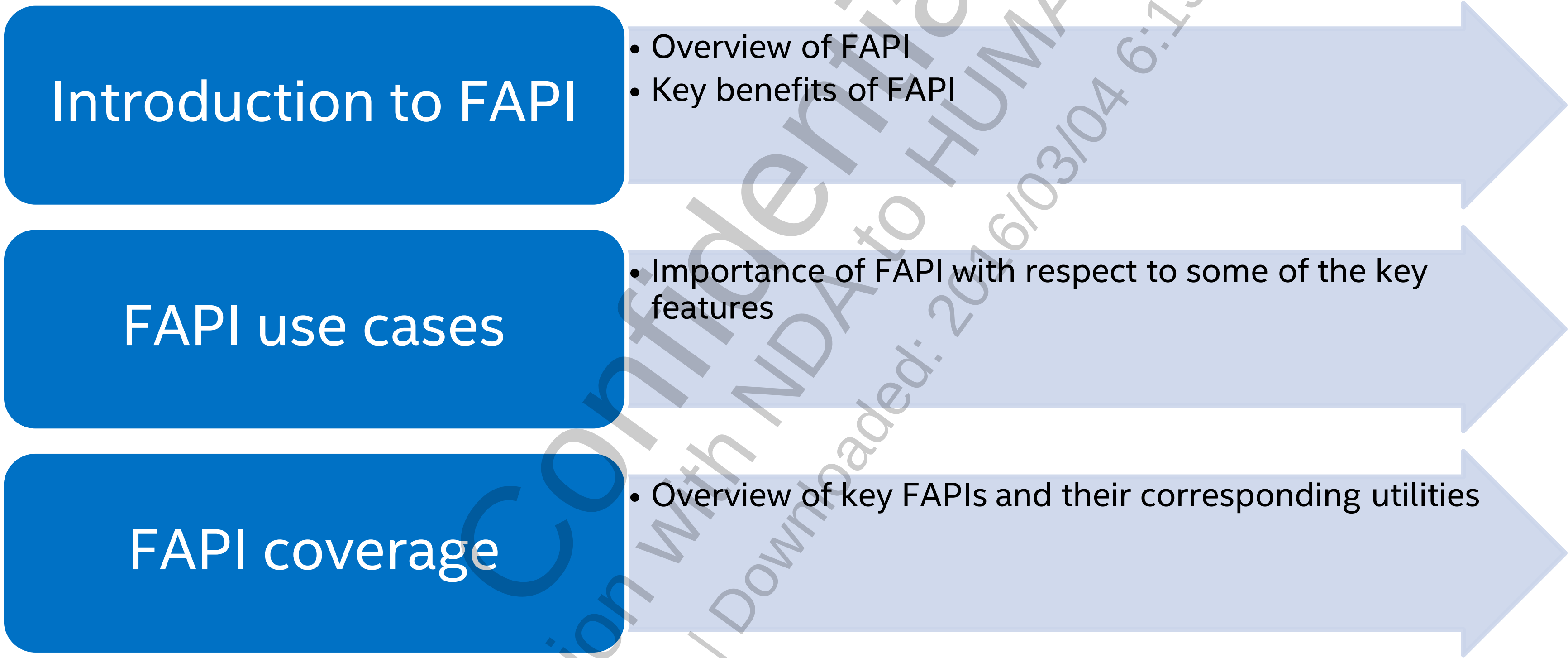
FAPI

14th Dec ,15

Connected Home Division

Intel Confidential

FAPI



Introduction to FAPI

- Overview of FAPI
- Key benefits of FAPI

FAPI use cases

- Importance of FAPI with respect to some of the key features

FAPI coverage

- Overview of key FAPIs and their corresponding utilities

FAPI overview

Functional APIs – core library for managing different software modules

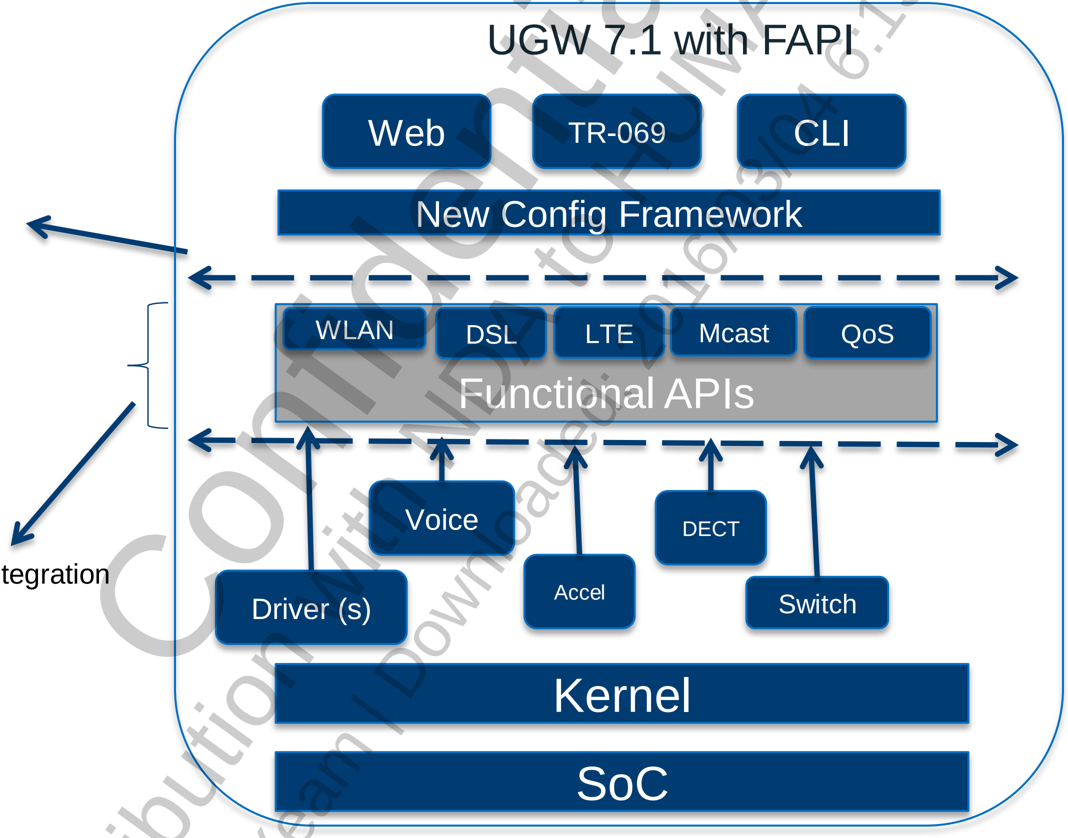
Generic APIs, hides HW complexities

Provides uniform interface across different SoC

Designed to be used “as-is”

Can be integrated with any framework

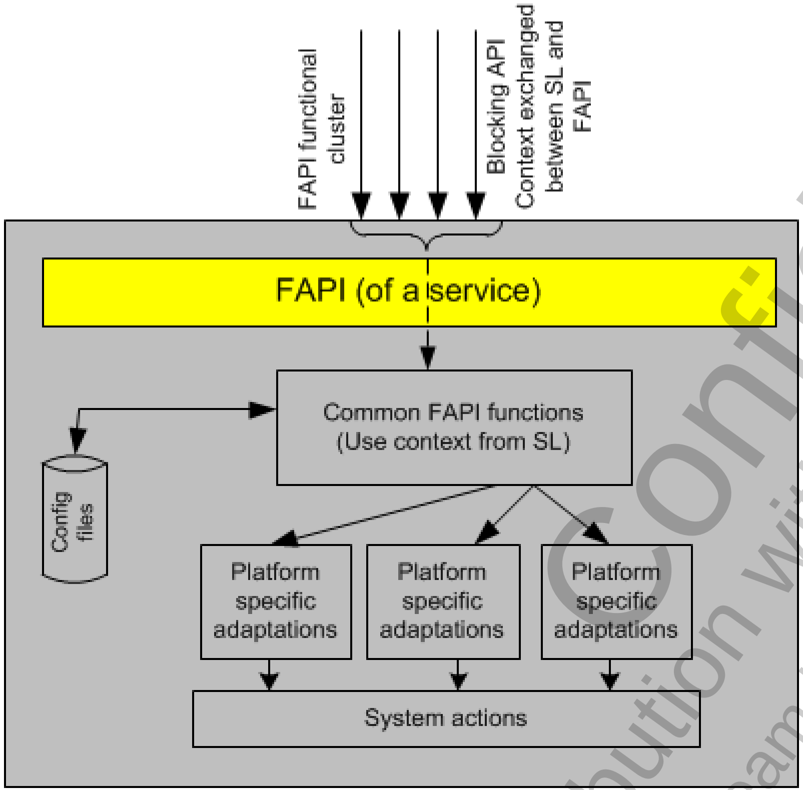
Functional API Layer



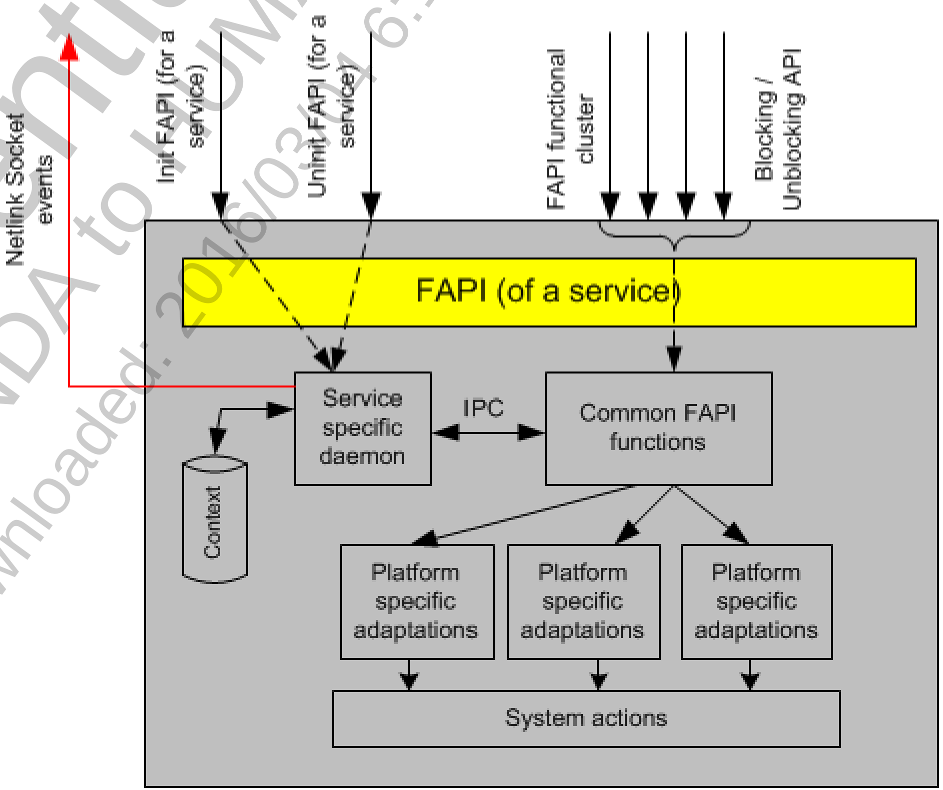
Interface used by customer

Reduce significant Integration effort for customer

FAPI – types

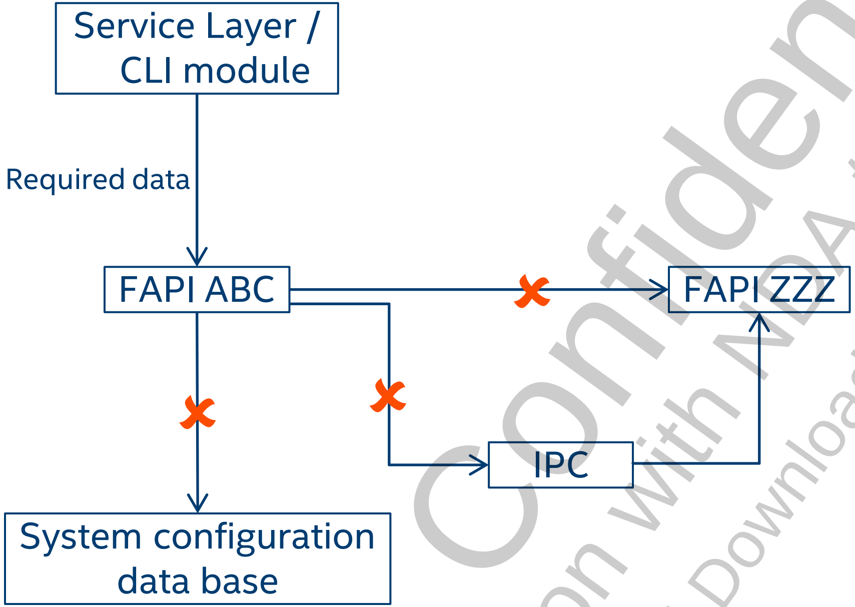


FAPI functional cluster
Blocking API
Context exchanged between SL and FAPI
FAPI (of a service)
Common FAPI functions (Use context from SL)
Config files
Platform specific adaptations
Platform specific adaptations
Platform specific adaptations
System actions



Netlink Socket events
Init FAPI (for a service)
Uninit FAPI (for a service)
FAPI functional cluster
Blocking / Unblocking API
FAPI (of a service)
Service specific daemon
IPC
Common FAPI functions
Context
Platform specific adaptations
Platform specific adaptations
Platform specific adaptations
System actions

FAPI – independent module



- ✓ No access to system configuration database
- ✓ No direct communication to other FAPI module
- ✓ No direct use of ubus or other framework specific messaging mechanisms

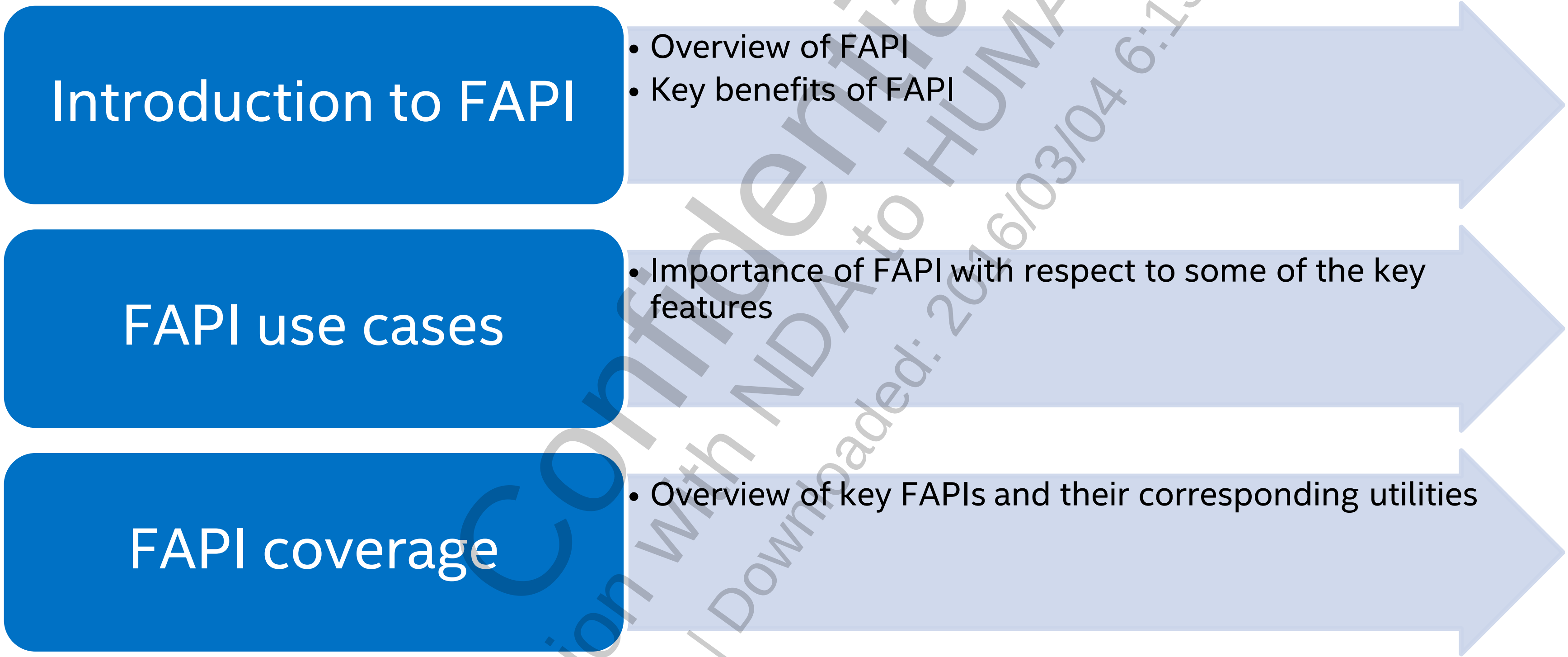
FAPI – modules in UGW-7.1

Module	FAPI location
DSL	/usr/lib/libdslfapi.so
QoS	/opt/lantiq/usr/lib/libqosfapi.so
LTE	/opt/lantiq/usr/lib/libwwanfapi.so
Multicast	/opt/lantiq/usr/lib/libmcastfapi.so
System	/usr/lib/libsysfapi.so /usr/lib/libethfapi.so

FAPI documentation

1. User’s Manual Programmer’s Reference document
2. User’s Manual Software Overview document

FAPI



Introduction to FAPI

- Overview of FAPI
- Key benefits of FAPI

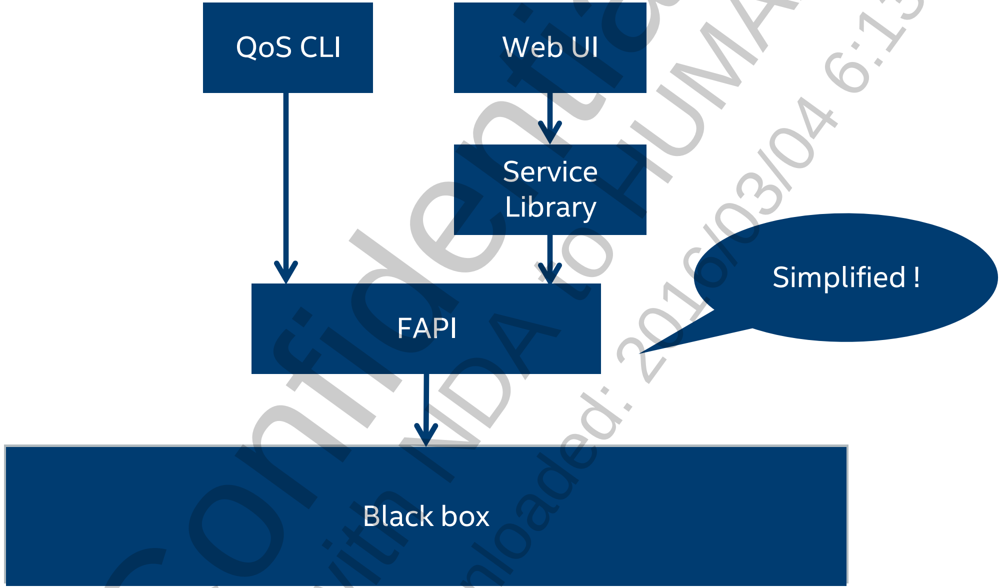
FAPI use cases

- Importance of FAPI with respect to some of the key features

FAPI coverage

- Overview of key FAPIs and their corresponding utilities

QoS FAPI

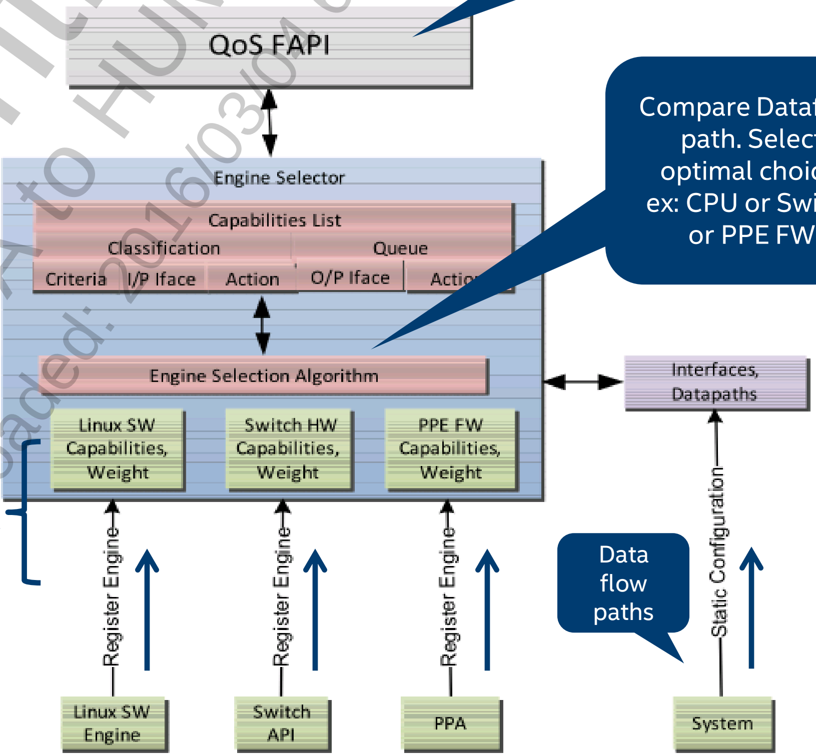


- Common APIs(FAPI) which abstracts HW/FW/SW
- Same FAPI across platforms
- Easy for customers to integrate / customize
- Better readability
- Easy to debug

QoS FAPI – framework

- Modular design
- Centralized QoS config mechanism
- Abstracts HW

fapi_qos_queue_set(….)
Ex: add queue for xDSL



Compare Dataflow path. Select optimal choice. ex: CPU or Switch or PPE FW

Data flow paths

- Registration of QoS capabilities from each engine
- Weightage of each engines (ex: Switch < PPE FW < Linux)

QoS FAPI – dependencies

PPA module

On GRX300 or VRX200 platforms, PPA modules to be reloaded to enable QoS

LAN and WAN switch port numbers

Switch configuration

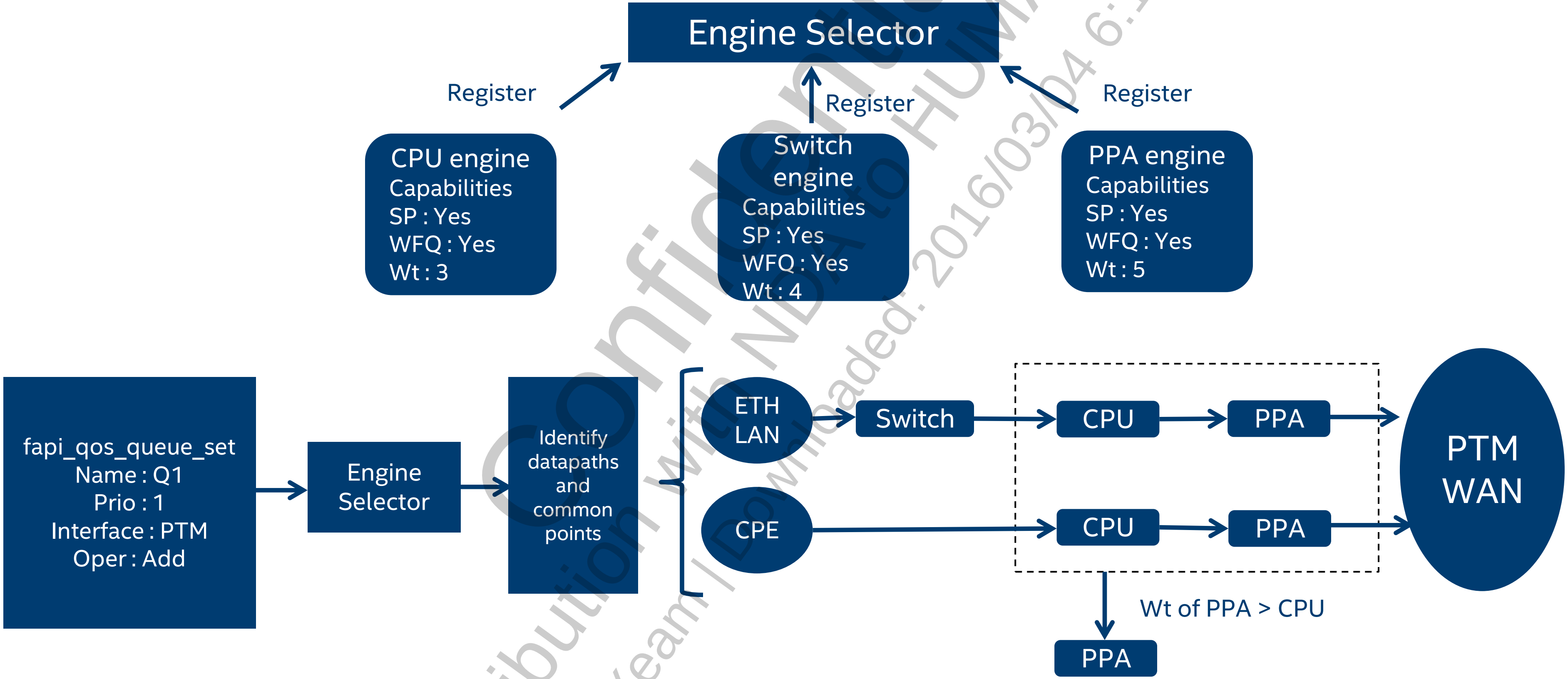
switch_cli utility or relevant ioctls

PPA configuration

ppacmd utility or relevant ioctls

XML parser library to parse platform configuration file

Illustration of Queue add operation



Engine Selector

Register

Register

Register

CPU engine
Capabilities
SP : Yes
WFQ : Yes
Wt : 3

Switch engine
Capabilities
SP : Yes
WFQ : Yes
Wt : 4

PPA engine
Capabilities
SP : Yes
WFQ : Yes
Wt : 5

fapi_qos_queue_set
Name : Q1
Prio : 1
Interface : PTM
Oper : Add

Engine Selector

Identify datapaths and common points

ETH LAN

CPE

Switch

CPU

PPA

CPU

PPA

PTM WAN

Wt of PPA > CPU

PPA

QoS FAPI – customizations

Adapter to interface with system

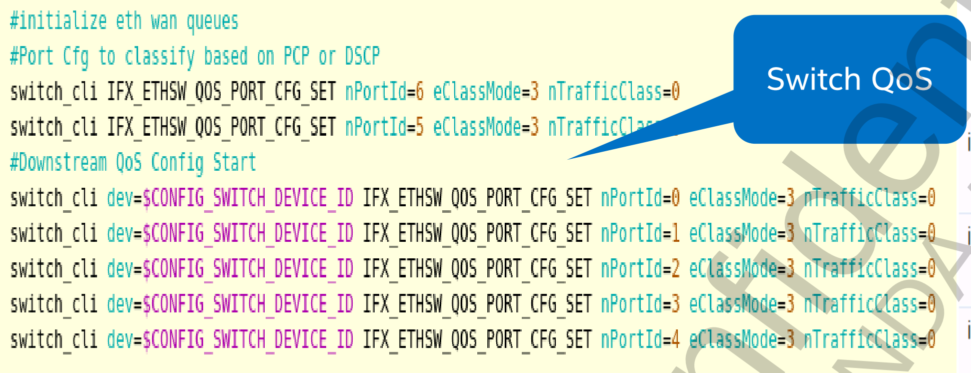
- Wrapper APIs are provided for adapters
- For instance, flexibility to use iptables command or relevant custom command

Default configuration of an engine

- Change in capabilities of appropriate engine.
  For instance, modify number of queues, enable ingress queuing support ..

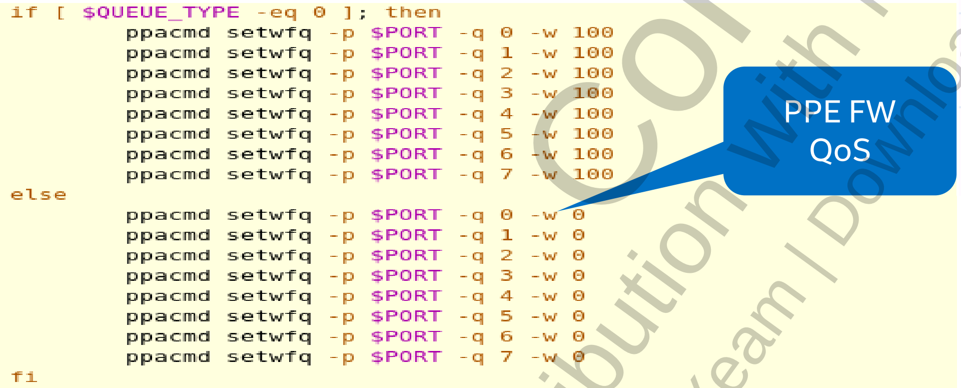
QoS – code comparison

Traditional approach



```
#initialize eth wan queues
#Port Cfg to classify based on PCP or DSCP
switch_cli IFX_ETHSW_QOS_PORT_CFG_SET nPortId=6 eClassMode=3 nTrafficClass=0
switch_cli IFX_ETHSW_QOS_PORT_CFG_SET nPortId=5 eClassMode=3 nTrafficCla
#Downstream QoS Config Start
switch_cli dev=$CONFIG_SWITCH_DEVICE_ID IFX_ETHSW_QOS_PORT_CFG_SET nPortId=0 eClassMode=3 nTrafficClass=0
switch_cli dev=$CONFIG_SWITCH_DEVICE_ID IFX_ETHSW_QOS_PORT_CFG_SET nPortId=1 eClassMode=3 nTrafficClass=0
switch_cli dev=$CONFIG_SWITCH_DEVICE_ID IFX_ETHSW_QOS_PORT_CFG_SET nPortId=2 eClassMode=3 nTrafficClass=0
switch_cli dev=$CONFIG_SWITCH_DEVICE_ID IFX_ETHSW_QOS_PORT_CFG_SET nPortId=3 eClassMode=3 nTrafficClass=0
switch_cli dev=$CONFIG_SWITCH_DEVICE_ID IFX_ETHSW_QOS_PORT_CFG_SET nPortId=4 eClassMode=3 nTrafficClass=0
```

Switch QoS

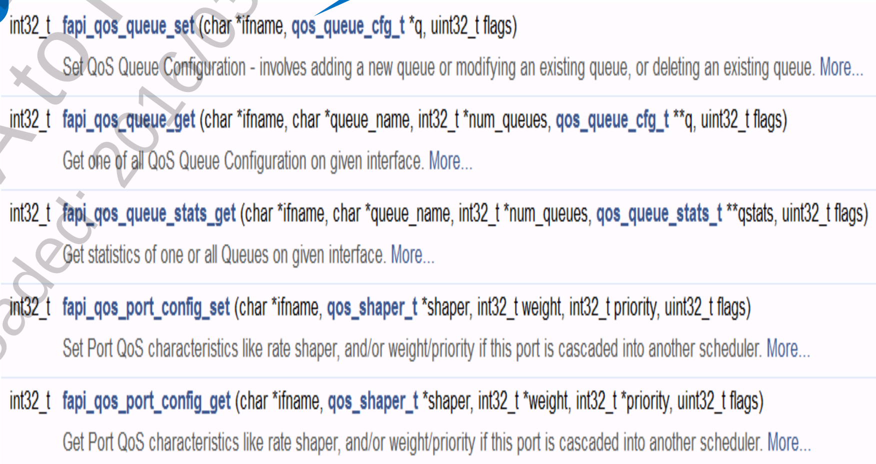


```
if [ $QUEUE_TYPE -eq 0 ]; then
        ppacmd setwfq -p $PORT -q 0 -w 100
        ppacmd setwfq -p $PORT -q 1 -w 100
        ppacmd setwfq -p $PORT -q 2 -w 100
        ppacmd setwfq -p $PORT -q 3 -w 100
        ppacmd setwfq -p $PORT -q 4 -w 100
        ppacmd setwfq -p $PORT -q 5 -w 100
        ppacmd setwfq -p $PORT -q 6 -w 100
        ppacmd setwfq -p $PORT -q 7 -w 100
else
        ppacmd setwfq -p $PORT -q 0 -w 0
        ppacmd setwfq -p $PORT -q 1 -w 0
        ppacmd setwfq -p $PORT -q 2 -w 0
        ppacmd setwfq -p $PORT -q 3 -w 0
        ppacmd setwfq -p $PORT -q 4 -w 0
        ppacmd setwfq -p $PORT -q 5 -w 0
        ppacmd setwfq -p $PORT -q 6 -w 0
        ppacmd setwfq -p $PORT -q 7 -w 0
fi
```

PPE FW QoS

UGW-7.1

Clean APIs – complexities hidden inside the APIs



int32_t **fapi_qos_queue_set** (char *ifname, qos_queue_cfg_t *q, uint32_t flags)

Set QoS Queue Configuration - involves adding a new queue or modifying an existing queue, or deleting an existing queue. More...

int32_t **fapi_qos_queue_get** (char *ifname, char *queue_name, int32_t *num_queues, qos_queue_cfg_t **q, uint32_t flags)

Get one of all QoS Queue Configuration on given interface. More...

int32_t **fapi_qos_queue_stats_get** (char *ifname, char *queue_name, int32_t *num_queues, qos_queue_stats_t **qstats, uint32_t flags)

Get statistics of one or all Queues on given interface. More...

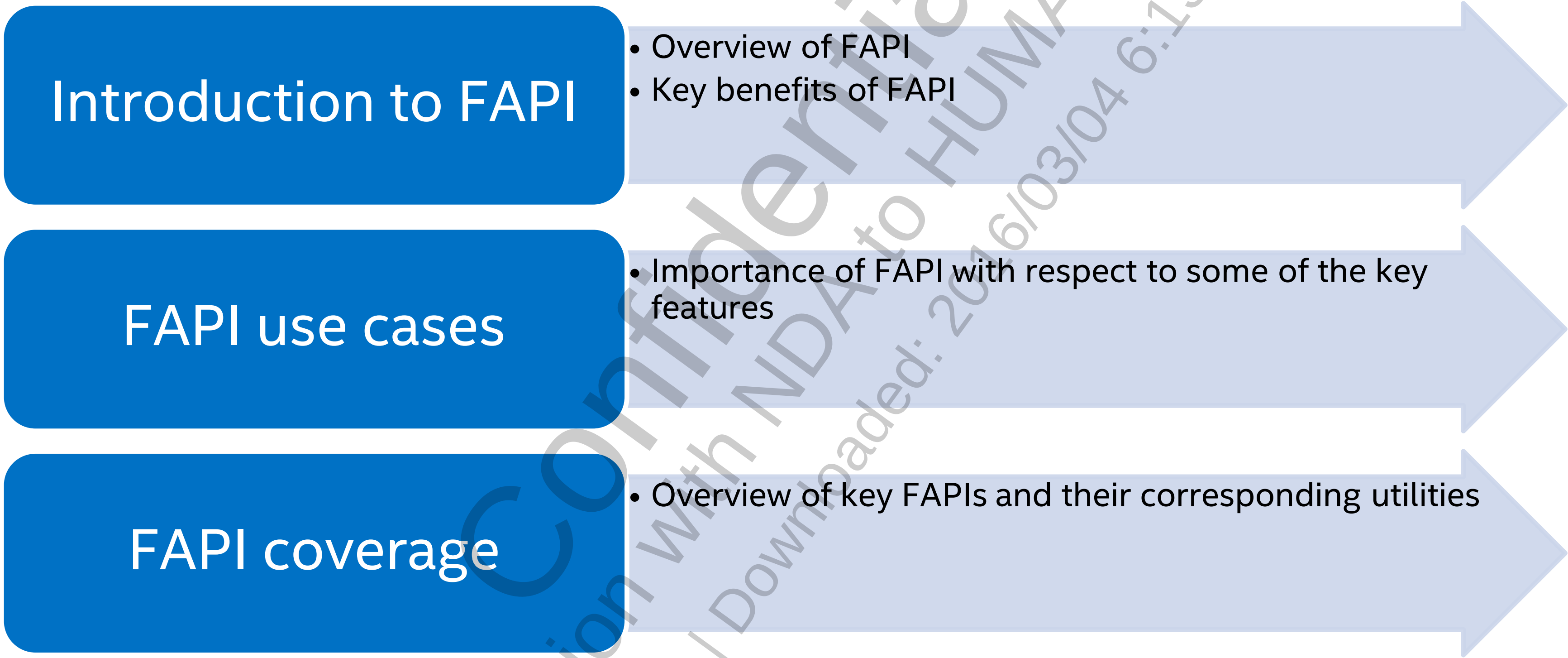
int32_t **fapi_qos_port_config_set** (char *ifname, qos_shaper_t *shaper, int32_t weight, int32_t priority, uint32_t flags)

Set Port QoS characteristics like rate shaper, and/or weight/priority if this port is cascaded into another scheduler. More...

int32_t **fapi_qos_port_config_get** (char *ifname, qos_shaper_t *shaper, int32_t *weight, int32_t *priority, uint32_t flags)

Get Port QoS characteristics like rate shaper, and/or weight/priority if this port is cascaded into another scheduler. More...

FAPI



Introduction to FAPI

- Overview of FAPI
- Key benefits of FAPI

FAPI use cases

- Importance of FAPI with respect to some of the key features

FAPI coverage

- Overview of key FAPIs and their corresponding utilities

FAPI coverage

Module	Description	CLI Utilities(s)	FAPI
System	System initialization, LAN port connectivity, Acceleration sub-system initialization, Switch statistics, etc…	sys_cli	fapi_eth_init fapi_sys_set, fapi_sys_get fapi_sys_if_attach, fapi_sys_if_detach fapi_get_portid fapi_port_set_status, fapi_port_get_status fapi_port_getbitrate, fapi_rmon_get fapi_port_setDuplexMode
QoS	QoS configuration – add/delete queue, add/delete classifier, etc…	qoscfg, ifcfg, qcfg, classcfg	fapi_qos_init fapi_qos_if_abs_set fapi_qos_if_abs_get fapi_qos_if_base_set fapi_qos_queue_set fapi_qos_classifier_set fapi_qos_port_config_get fapi_qos_port_config_set
Multicast	Multicast group membership management – add groups to Acceleration, Switch, etc…	mcast_cli	fapi_mch_add_entry fapi_mch_update_entry fapi_mch_del_entry

Contd.,

Module	Description	CLI Utilities(s)	FAPI
LTE	LTE module configuration and status check, etc…	wwan_cli	wwanfapi_do_init wwanfapi_do_connect wwanfapi_do_disconnect wwanfapi_get_interfacestatus wwanfapi_get_signalstrength wwanfapi_get_usimstatus wwanfapi_do_setpin wwanfapi_do_unlock wwanfapi_at_set
DSL	DSL initialization, status check, statistics, etc…	dsl_fapid	fapi_dsl_open fapi_dsl_init fapi_dsl_close fapi_dsl_uninit fapi_dsl_line_get fapi_dsl_line_set fapi_dsl_channel_stats_showtime_get fapi_dsl_channel_stats_showtime_set fapi_dsl_channel_stats_last_showtime_get fapi_dsl_channel_stats_last_showtime_set fapi_dsl_channel_stats_current_day_get fapi_dsl_channel_stats_current_day_set and more

intel®

experience
what's inside™

Backup

Motivation

- In UGW, QoS flows are influen p AN mode, platform, acceleration req
- For each flow, rent
- Understandin mework – web, scripts. A

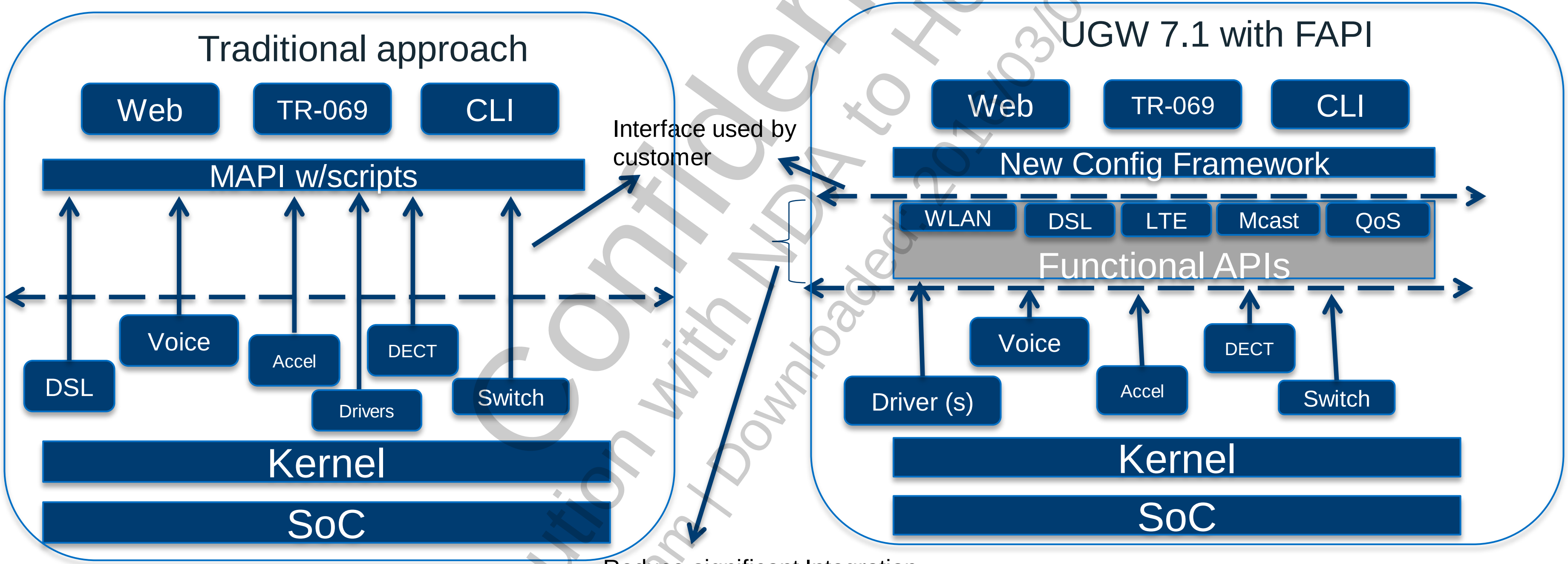
Tool that,

- Showcases QoS capabilities in simple manner
- Doesn’t use web, scripts from QoS infrastructure
- Enables customers to understand and use our QoS infrastructure effectively

	XRX2					
	Cla				n	Queuing
ADSL xTM WAN	Softw				ftware	Software – 8 Qs
Eth WAN	Software			n – 8 Qs	Switch	Switch – 8 Qs
VDSL PTM WAN	Software	Firmware –	Softw	Firmware – 8 Qs	Software	Firmware – 8 Qs
Downstream						
xDSL or Eth WAN	Switch	Switch – 4 Qs	Switch	Switch – 4 Qs	Switch	Switch – 4 Qs

Functional API Layer

- Higher Layer Functional API
  - Reduces the complexity for customer to plug-in their services

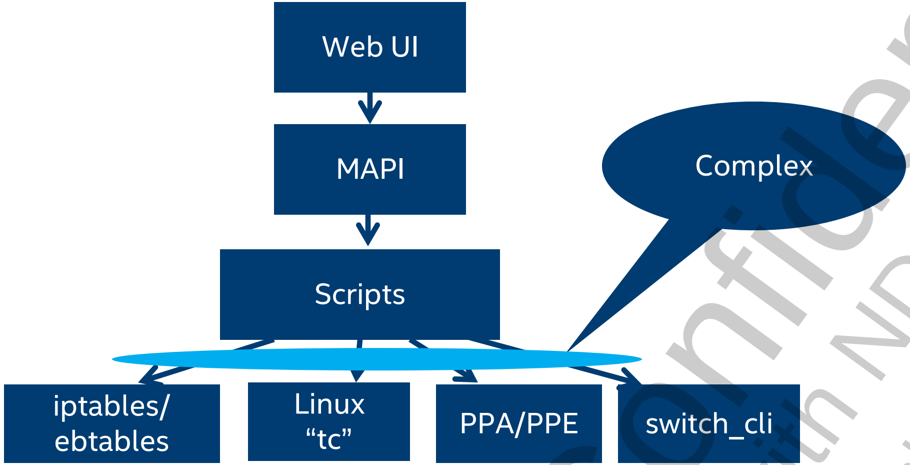


Interface used by customer

Reduce significant Integration effort for customer

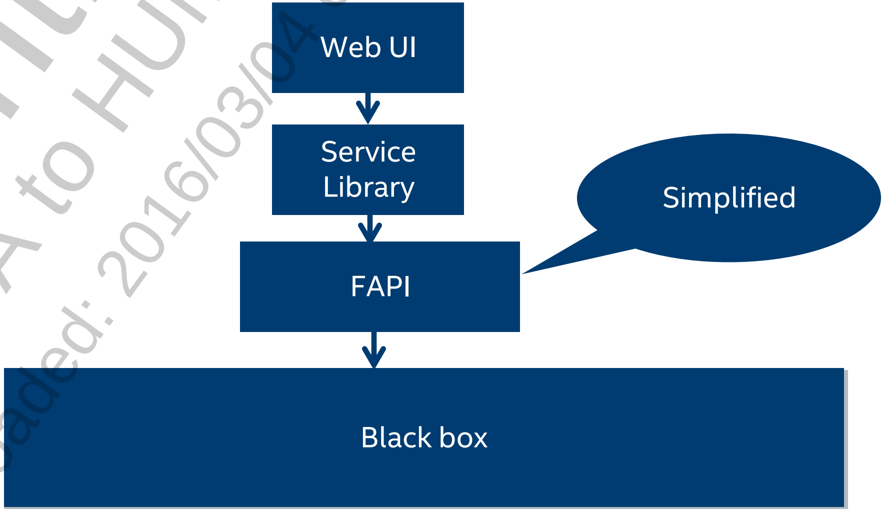
QoS FAPI – comparison

Traditional approach



- Manage multiple configuration objects - complex
- More time to adapt to New platforms
- Difficult for customers to integrate / customize
- Low readability
- Difficult to debug

UGW-7.1



- Common APIs(FAPI) which abstracts HW/FW/SW
- Same FAPI across platforms
- Easy for customers to integrate / customize
- Better readability
- Easy to debug