Linux firmware for iRMC controller on Fujitsu Primergy servers

Vladimir Shakhov
R&D Senior Development Engineer

Fujitsu Technology Solutions, R&D department

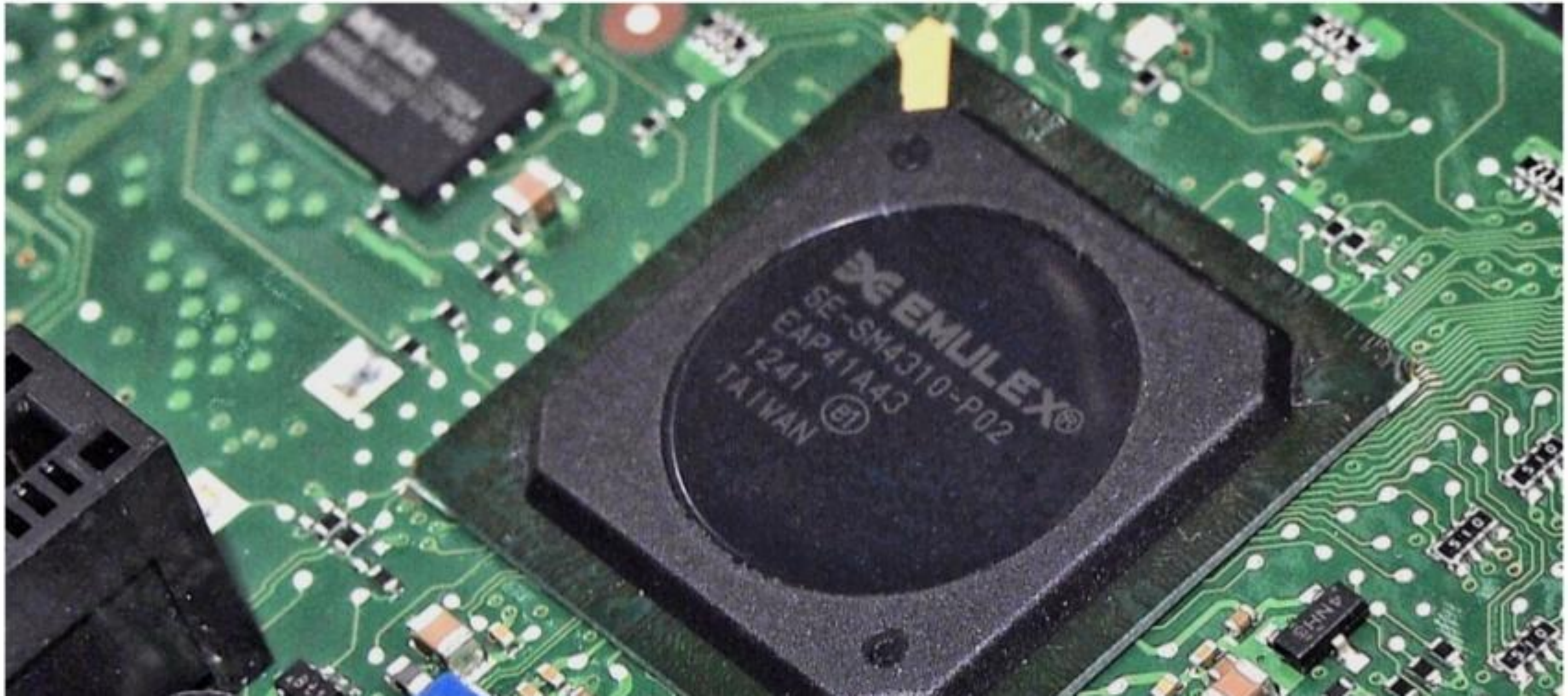
Table of content

1. **Fujitsu Primergy servers family**
   - What is iRMC
   - Software - ServerView Suite
   - Open standards: IPMI protocol and others
   - iRMC internals
   - Demo: WebIF and IPMI
2. **Road to Linux: from iRMC S1/ThreadX to iRMC S4/Linux**
   - Early days - ThreadX : S1 - S2/S3
   - Migration to Linux: S4
   - Demo: RemoteManager - bug-to-bug compatible

Fujitsu Primergy Servers

Lineage of x86-based servers:
Blade (BX), Rack (RX), Tower (TX) and Cloud (CX).

iRMC S4 in the wild



iRMC - integrated Remote Management Controller

ARM-based SoC

Emulex Pilot3 iBMC ASIC

Integrated BMC

Super I/O

Graphics controller

KVMS: Remote Keyboard, Video, Mouse and Storage

CPU: 32-bit 400MHz ARM9 processor with MMU.

iRMC - integrated Remote Management Controller

ARM-based SoC

Emulex Pilot3 iBMC ASIC

Integrated BMC

Super I/O

Graphics controller

KVMS: Remote Keyboard, Video, Mouse and Storage

CPU: 32-bit 400MHz ARM9 processor with MMU.

Control x86 hardware

Work independent if x86 host on or off

iRMC - integrated Remote Management Controller

ARM-based SoC

Emulex Pilot3 iBMC ASIC

Integrated BMC

Super I/O

Graphics controller

KVMS: Remote Keyboard, Video, Mouse and Storage

CPU: 32-bit 400MHz ARM9 processor with MMU.

Control x86 hardware

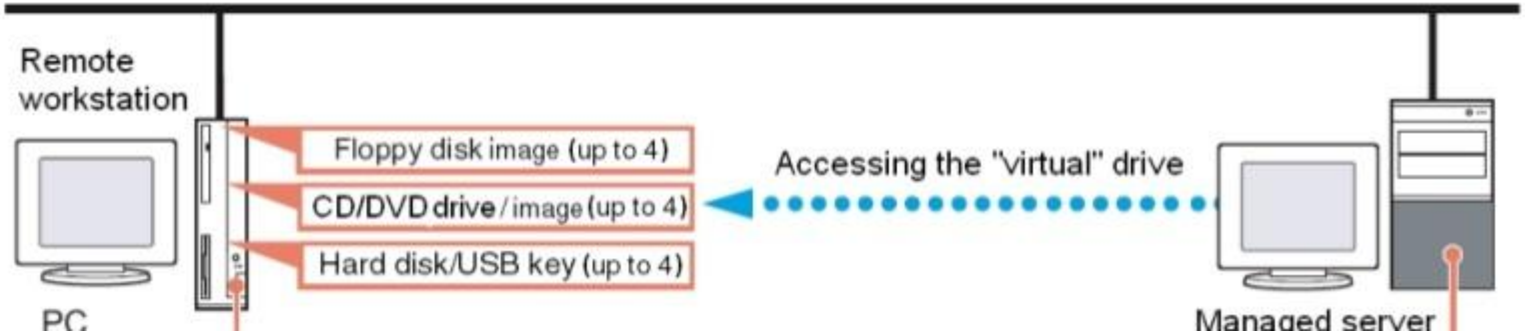
Work independent if x86 host on or off

iRMC basic features

- Web access (own web-server)
- Security (SSL and SSH included)
- ServerView suite Integration
- Power management
- SNMPv1/v2c/v3 support
- Text console redirection
- “Headless” system operation

iRMC advanced features

- Advanced Video Redirection (AVR)
- Virtual Media
- Embedded Lifecycle Management (eLCM)


Remote workstation
Floppy disk image (up to 4)
CD/DVD drive / image (up to 4)
Hard disk/USB key (up to 4)
Accessing the "virtual" drive
PC
Managed server

Open standards

Intelligent Platform Management Interface

IPMI - standardized, abstract, message-based interface between BMC and intelligent hardware for platform management. Key component of system.

Open standards

Intelligent Platform Management Interface

IPMI - standardized, abstract, message-based interface between BMC and intelligent hardware for platform management. Key component of system.

Web: HTTP, HTML, JavaScript

Web-based control interface.

Open standards

Intelligent Platform Management Interface

IPMI - standardized, abstract, message-based interface between BMC and intelligent hardware for platform management. Key component of system.

Web: HTTP, HTML, JavaScript

Web-based control interface.

SNMP ver 1/2x/3

Open standards

Intelligent Platform Management Interface

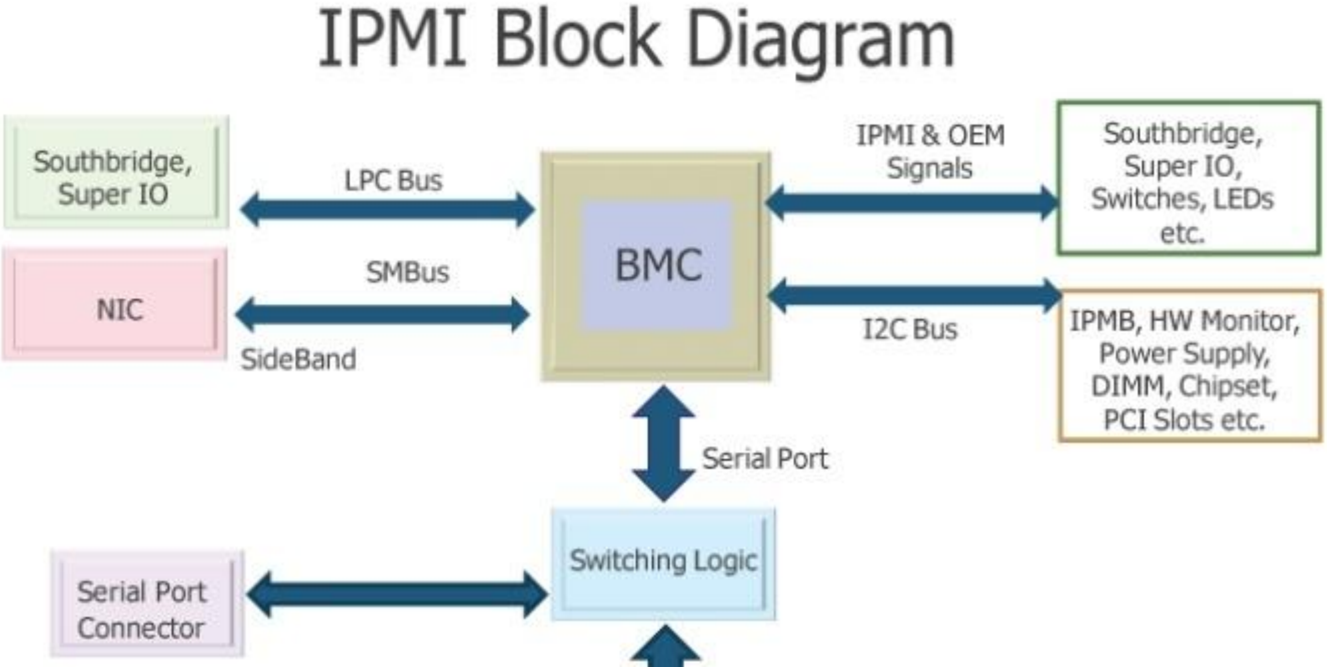
IPMI - standardized, abstract, message-based interface between BMC and intelligent hardware for platform management. Key component of system.

Web: HTTP, HTML, JavaScript

Web-based control interface.

SNMP ver 1/2x/3

IPMI - key interface of a system



Demo 1

Web interface: AVR, VirtualMedia, remote boot

Scenario 1: AVR, show boot settings

AVR: show Windows, Start LCM Custom Image, AVR: Show Linux

Demo 1

Web interface: AVR, VirtualMedia, remote boot

Scenario 1: AVR, show boot settings

AVR: show Windows, Start LCM Custom Image, AVR: Show Linux

Scenario 2: IPMI - via ipmitool

```
$ ipmitool -U admin -P admin -H 192.168.1.1 -I lanplus [command line]
```

command line variants:

- chassis status
- lan print

iRMC S1 - S2/S3 OS

THREADX®

Pro

- Advanced Real-Time Operation System
- Small footprint
- Fast performance

iRMC S1 - S2/S3 OS

THREADX®

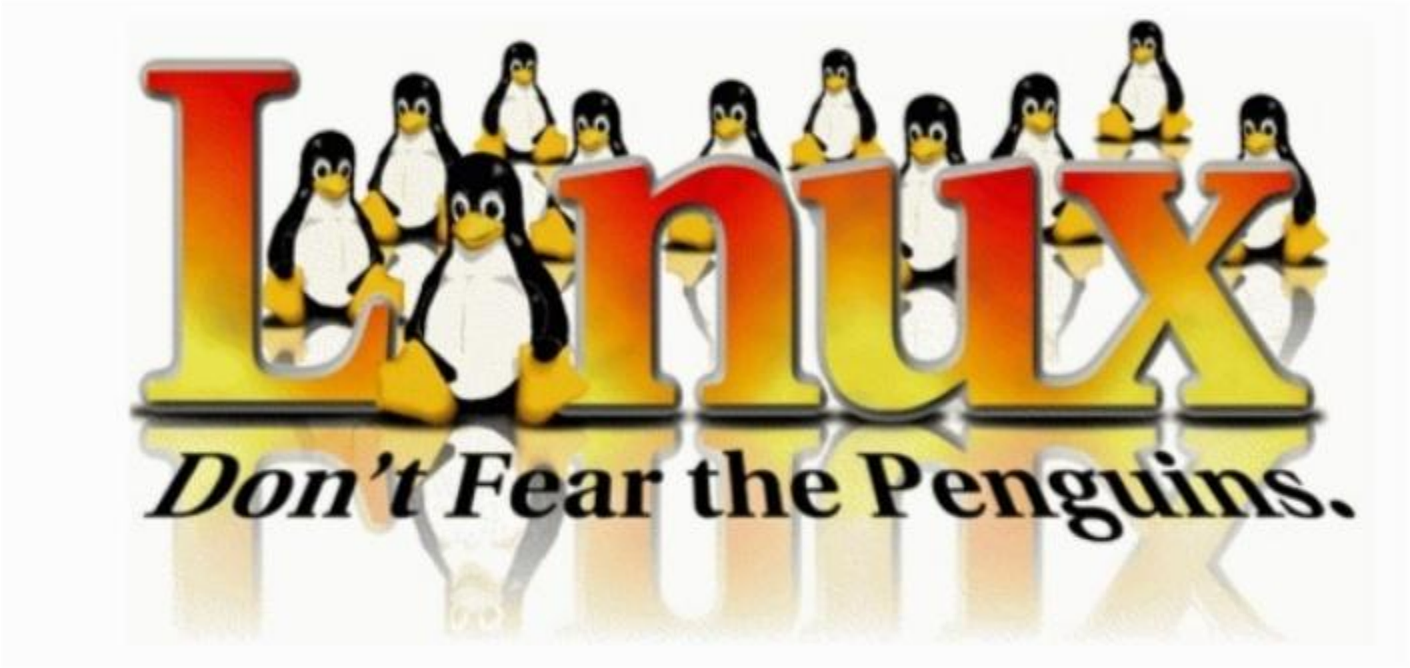
Pro

- Advanced Real-Time Operation System
- Small footprint
- Fast performance

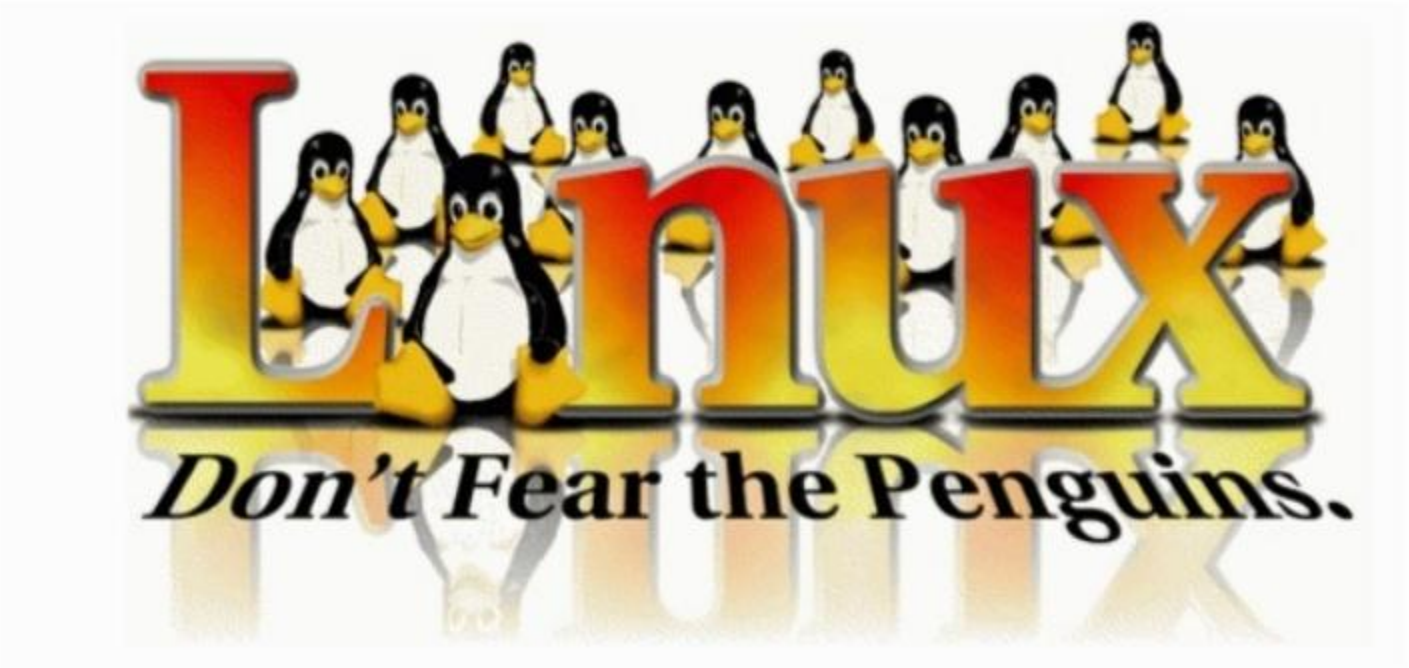
Contra

- Lack of available developers

Why Linux



Why Linux



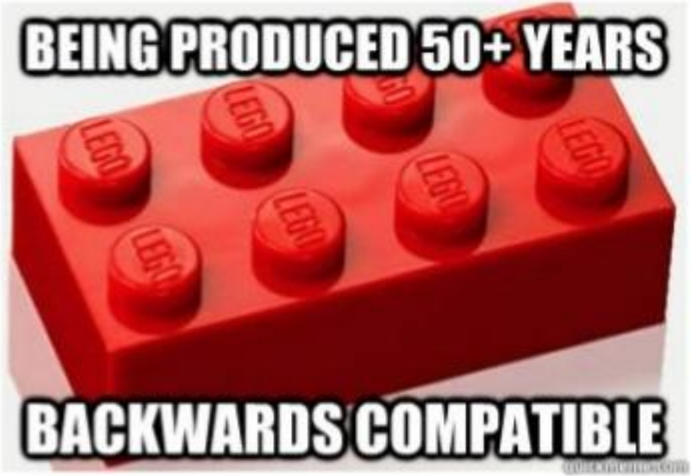
Cost of development and support

- More developers available

Main challenges

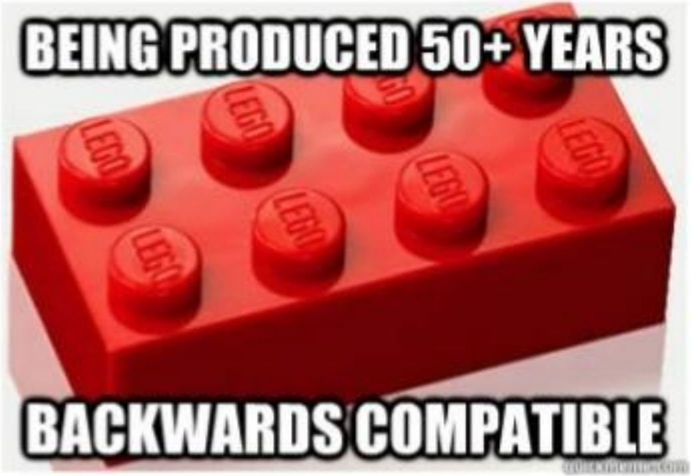
Backward compatibility

- Same interfaces (UI, protocols)
- Binary firmware upgrades



Main challenges

Backward compatibility



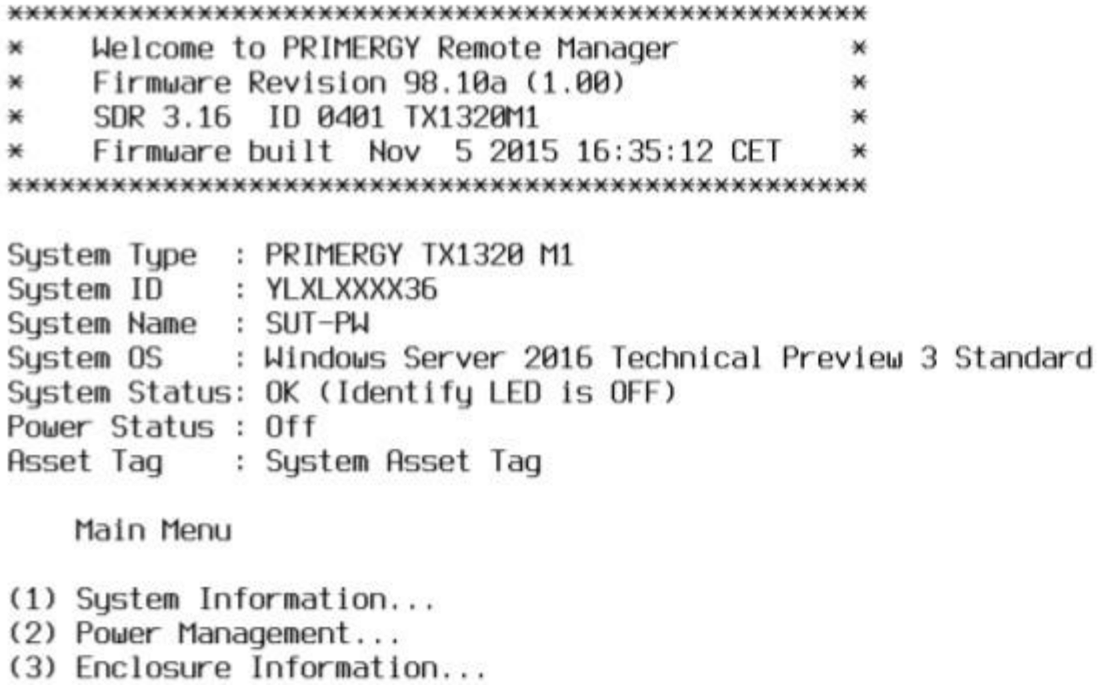
- Same interfaces (UI, protocols)
- Binary firmware upgrades

Code re-use

OS APIs are different

Demo 2: OpenSSH + RemoteManager

Interface ~~bug to bug~~ byte to byte identical to ThreadX.



```
**************************************************
*    Welcome to PRIMERGY Remote Manager          *
*    Firmware Revision 98.10a (1.00)             *
*    SDR 3.16  ID 0401 TX1320M1                  *
*    Firmware built  Nov  5 2015 16:35:12 CET    *
**************************************************

System Type  : PRIMERGY TX1320 M1
System ID    : YLXLXXXX36
System Name  : SUT-PW
System OS    : Windows Server 2016 Technical Preview 3 Standard
System Status: OK (Identify LED is OFF)
Power Status : Off
Asset Tag    : System Asset Tag

    Main Menu

(1) System Information...
(2) Power Management...
(3) Enclosure Information...
```

iRMC Firmware components

Free and Open Source Software

- Linux Kernel
- U-Boot bootloader
- Busybox
- GNU Glibc
- Net-SNMP
- OpenSSH

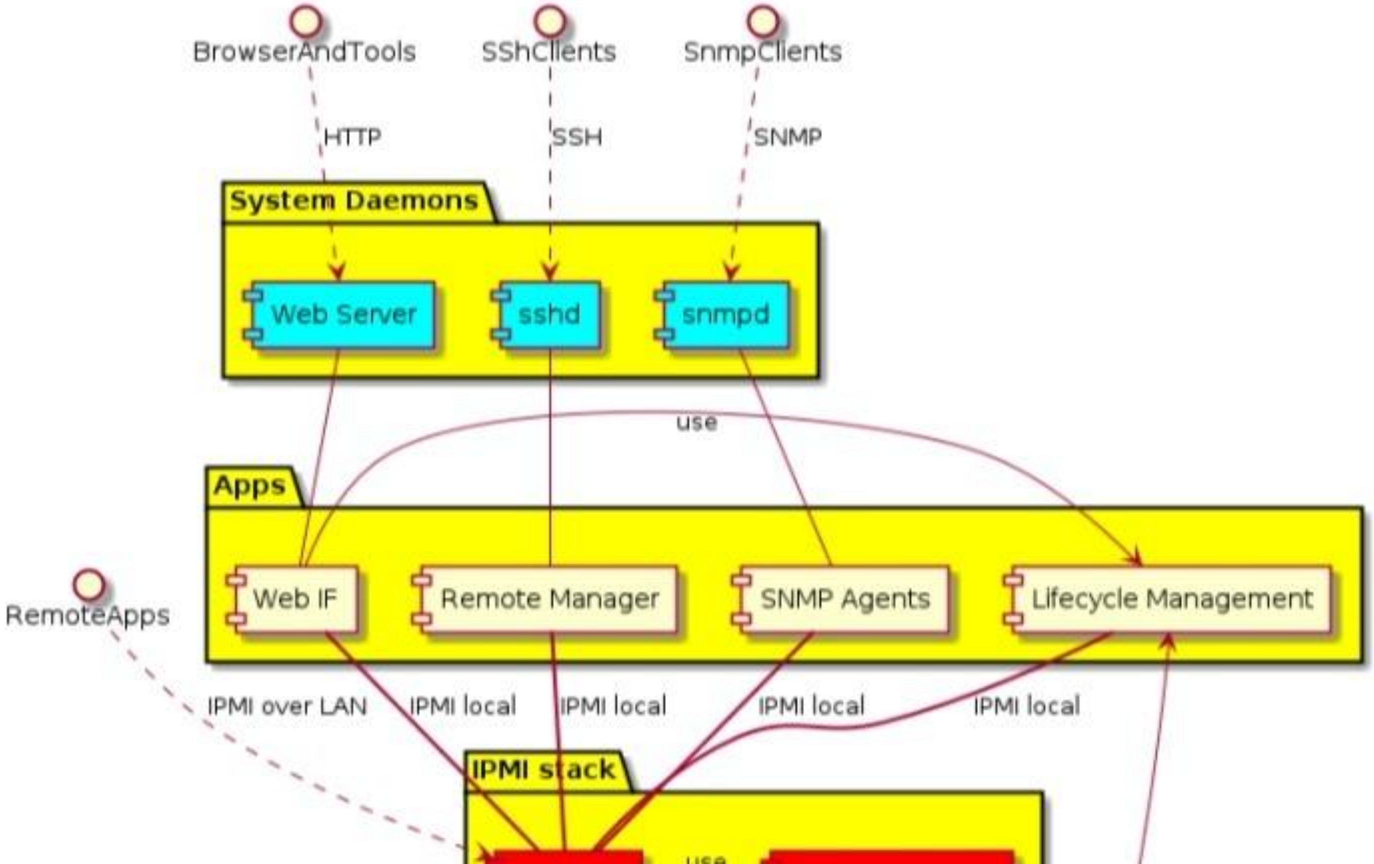
iRMC Firmware components

Free and Open Source Software

- Linux Kernel
- U-Boot bootloader
- Busybox
- GNU Glibc
- Net-SNMP
- OpenSSH

Closed source

iRMC firmware internals



Development environment

LXC containers + X2go for developers

The same environment for all to build and debug.

Read-only root filesystem on container.

Debian GNU/Linux based.

Custom package system

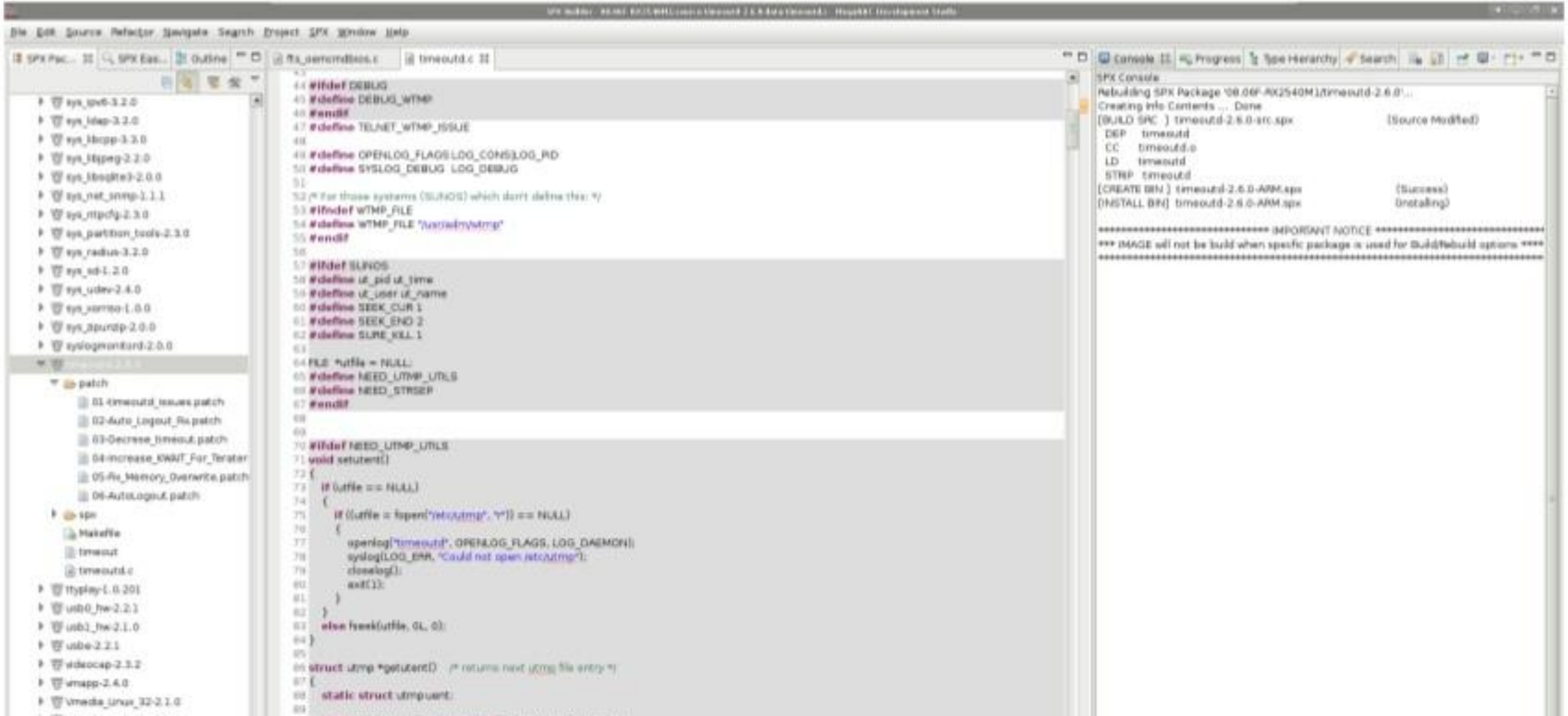
Used only in development and build process.

Not used for updates.

Package format similar to DEB, but not the same.

Eclipse-based IDE

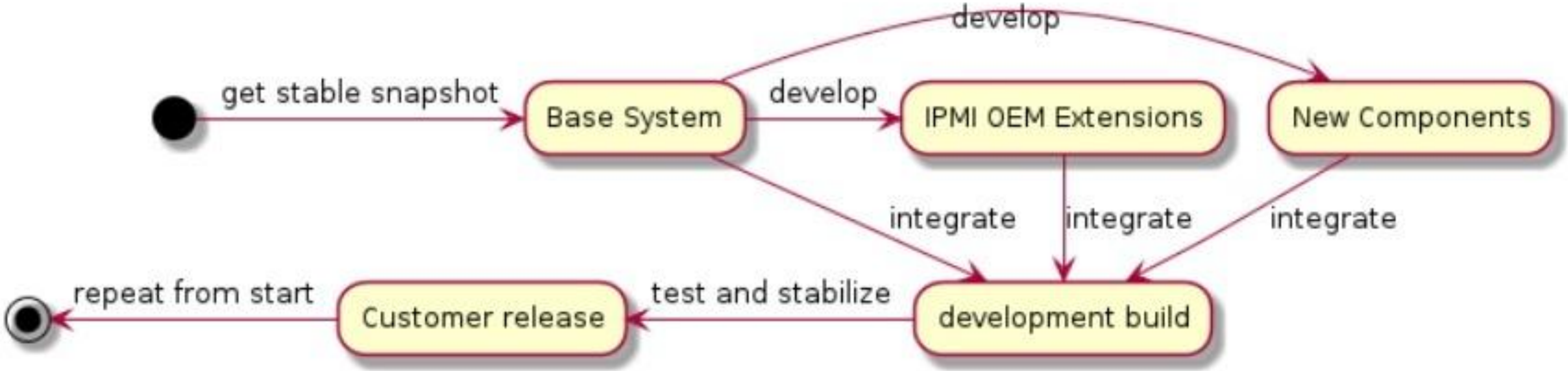
IDE



```c
#ifdef DEBUG
#define DEBUG_WTMP
#endif
#define TELNET_WTMP_ISSUE

#define OPENLOG_FLAGS LOG_CONS|LOG_PID
#define SYSLOG_DEBUG  LOG_DEBUG

/* For those systems (SUNOS) which don't define this: */
#ifndef WTMP_FILE
#define WTMP_FILE "/usr/adm/wtmp"
#endif

#ifdef SUNOS
#define ut_pid ut_time
#define ut_user ut_name
#define SEEK_CUR 1
#define SEEK_END 2
#define SURE_KILL 1

FILE *utfile = NULL;
#define NEED_UTMP_UTILS
#define NEED_STRSEP
#endif


#ifdef NEED_UTMP_UTILS
void setutent()
{
  if (utfile == NULL)
  {
    if ((utfile = fopen("/etc/utmp", "r")) == NULL)
    {
      openlog("timeoutd", OPENLOG_FLAGS, LOG_DAEMON);
      syslog(LOG_ERR, "Could not open /etc/utmp");
      closelog();
      exit(1);
    }
  }
  else fseek(utfile, 0L, 0);
}

struct utmp *getutent()   /* returns next utmp file entry */
{
  static struct utmp uent;
```

```
SPX Console
Rebuilding SPX Package '08.06F-RX2540M1/timeoutd-2.6.0'...
Creating Info Contents ...  Done
[BUILD SRC  ] timeoutd-2.6.0-src.spx                    (Source Modified)
  DEP    timeoutd
  CC     timeoutd.o
  LD     timeoutd
  STRIP  timeoutd
[CREATE BIN ] timeoutd-2.6.0-ARM.spx                    (Success)
[INSTALL BIN] timeoutd-2.6.0-ARM.spx                    (Installing)

******************************** IMPORTANT NOTICE ********************************
*** IMAGE will not be build when specfic package is used for Build/Rebuild options ****
**********************************************************************************
```

Development cycle



Very typical Embedded Linux development cycle (simplified view):

1. Get base system snapshot and freeze it
2. Develop new components and IPMI OEM extensions

FOSS legal questions

- Following the FOSS licenses
- Special policy for FOSS components using
- Consolidation of components legal status
- Rare upstream communication [1]
- FOSS component sources - by demand from support

Demo 3

Development login via SSH.
Show typical Embedded Linux system.

Questions? Remarks?

FUJITSU

shaping tomorrow with you

- Fujitsu Primergy servers: http://www.fujitsu.com/fts/products/computing/servers/primergy/
- iRMC S4 manual: http://manuals.ts.fujitsu.com/file/11470/irmc-s4-ug-en.pdf
- Emulex Pilot 3 iBMC specs: http://www.emulex.com/products/controllers/management-controllers/pilot-