# Journaling Filesystem

**VA Linux Systems Japan**

岡島順治郎

**jro@valinux.co.jp**





VA LINUX SYSTEMS JAPAN

# ファイル生成時間を計る time(bash)

```
#!/bin/bash -
n=${1:-100}
while test $n -gt 0; do s="$s $n"; let n-=1; done
time touch $s
rm $s
```

| | ext2fs | Another ext2fs |
|---|---|---|
| real | 0m0.055s | 0m2.184s |
| user | 0m0.000s | 0m0.000s |
| sys | 0m0.010s | 0m0.000s |

# ファイル生成時間を計る iostat(1) - 1

```
#!/bin/bash -
n=${1:-100}
while test $n -gt 0; do s="$s $n"; let n-=1; done
set -m
iostat -d 1 &
touch $s
sleep 1
kill %iostat
rm $s
```

# ファイル生成時間を計る iostat(1) – 2

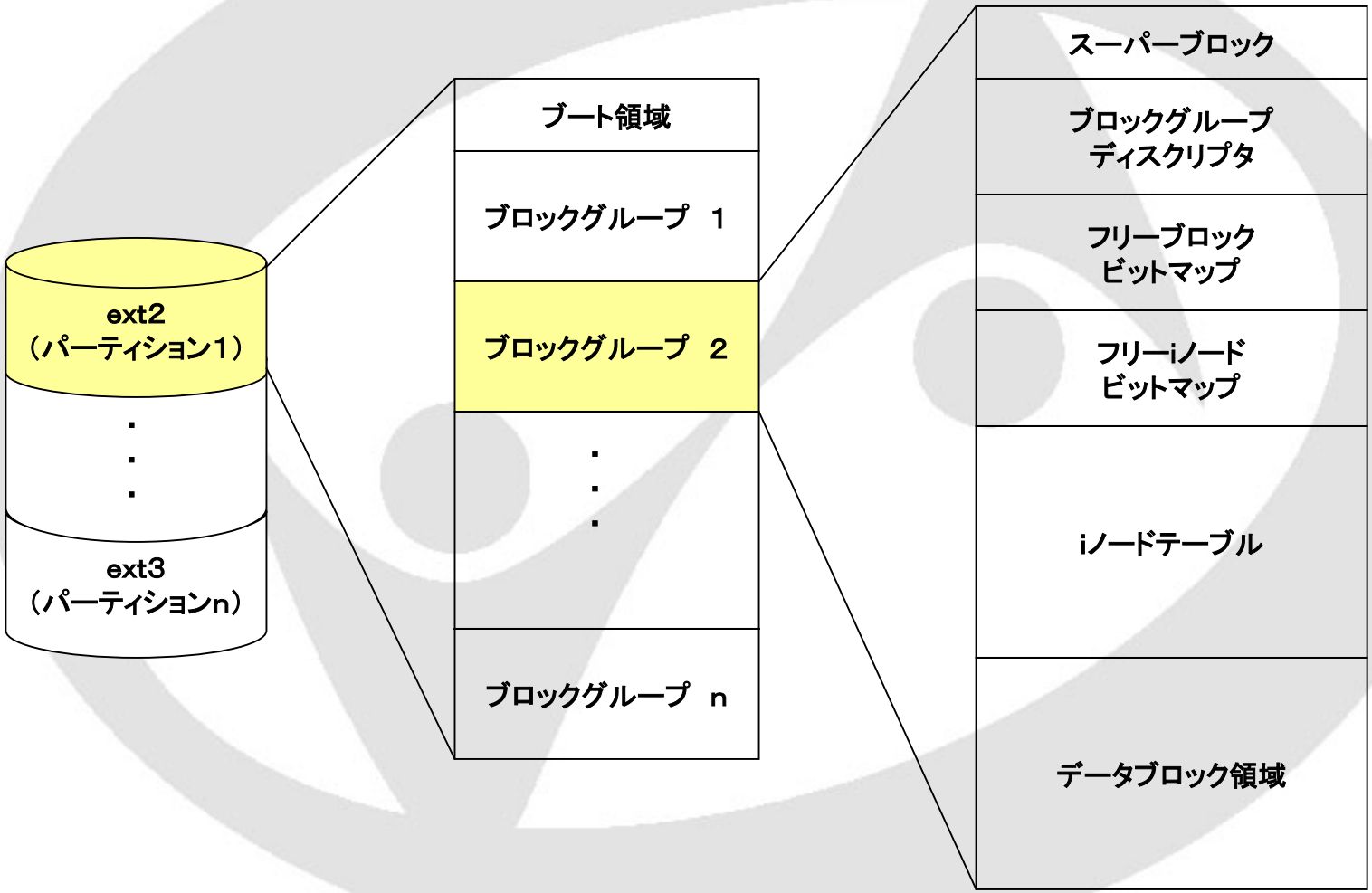
| ext2fs | tps | Bread/s | Bwrite/s | Bread | Bwrtn |
|---|---|---|---|---|---|
| | 0.00 | 0.00 | 0.00 | 0 | 0 |
| Another ext2fs | tps | Bread/s | Bwrite/s | Bread | Bwrtn |
| | 120.00 | 0.00 | 960.00 | 0 | 960 |
| | 115.00 | 0.00 | 928.00 | 0 | 928 |
| | 113.00 | 0.00 | 904.00 | 0 | 904 |
| | 116.00 | 0.00 | 936.00 | 0 | 936 |

# Mount option, sync と async

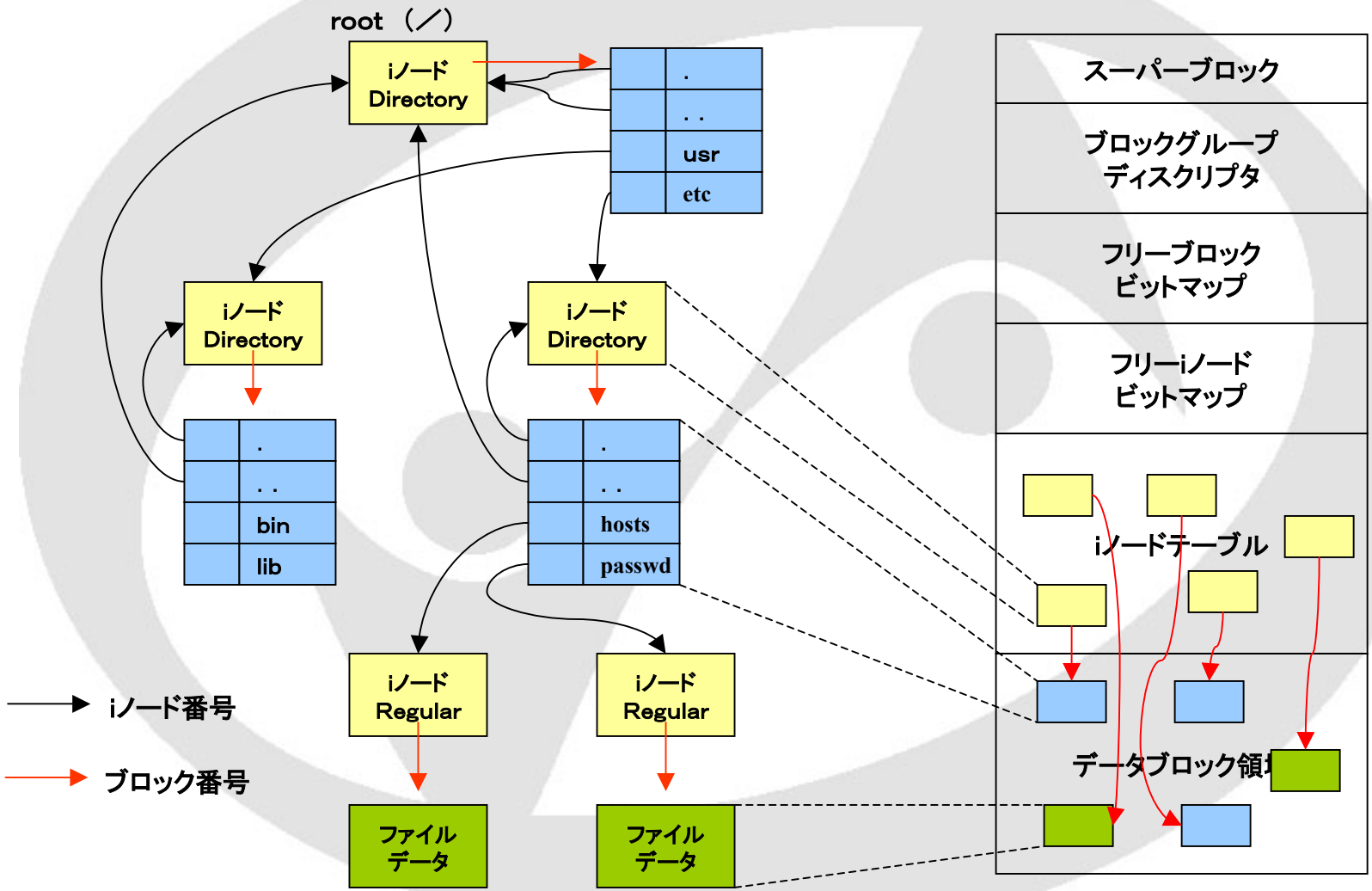
touch(1)の例からわかること

| | |
|---|---|
| sync option | open(O_CREAT)の最中に何かを大量に書き込み、時間がかかる。 |
| async option | 基本的に書き込みを行わないため、時間がかからない。<br><shameless plug><br><a href="http://www03.u-page.so-net.ne.jp/da2/h-takaha/"> |
| open(2)の O_SYNC | open(2)後のwrite(2)に作用する。touch(1)によるファイル作成ではファイルの中身(ファイルデータ)は書き込んでいない。 |

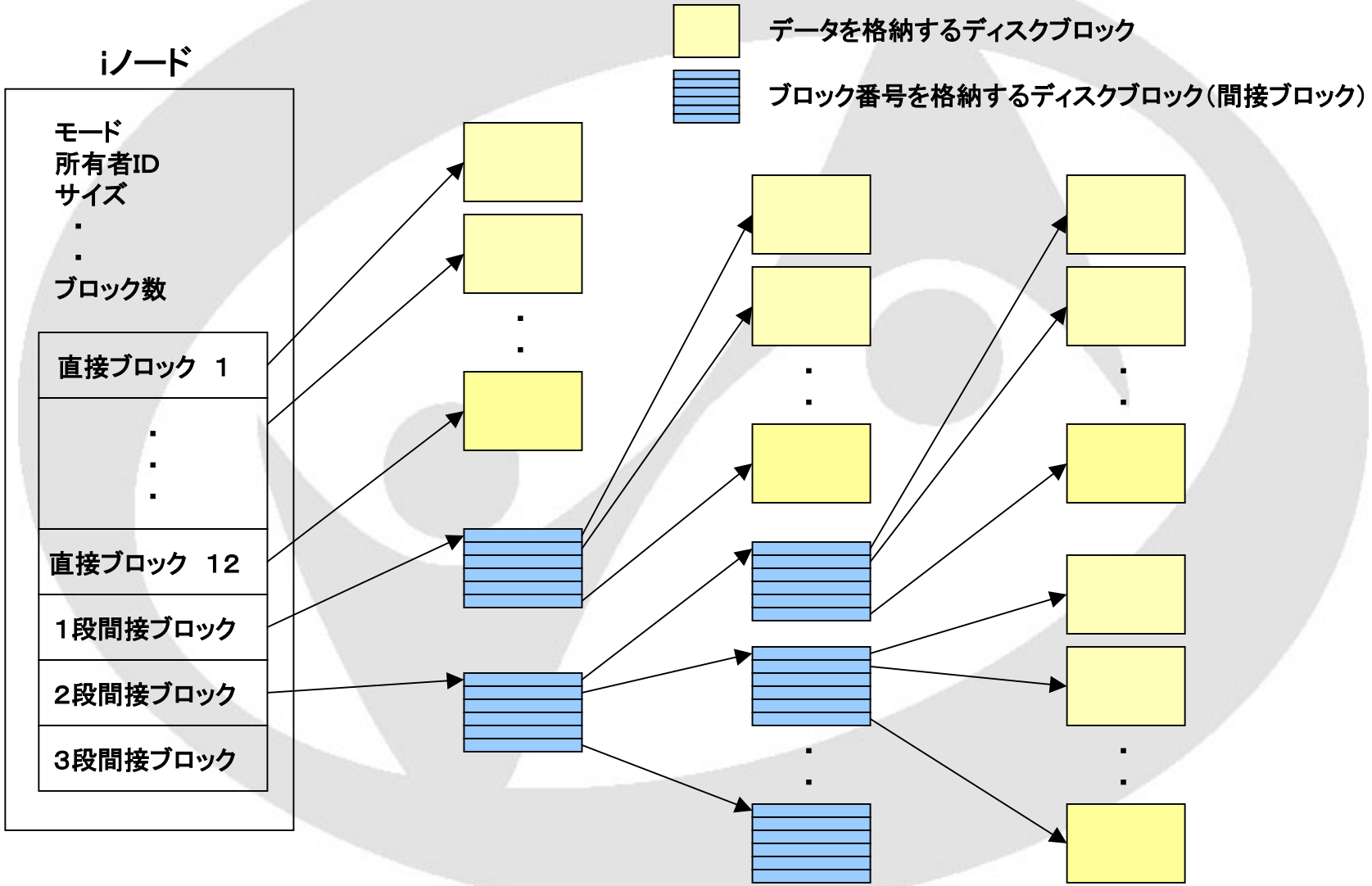
# ext2：レイアウト



# ext2：ディレクトリ階層



# ext2：ファイルデータ構造








VA LINUX SYSTEMS JAPAN

# メタデータ

ファイルシステム上でファイルの中身(ファイルデータ)以外の情報。
touch(1)の例から取り上げるものは、以下の三つ。

| | |
|---|---|
| 親ディレクトリブロック | ファイル名とinode番号の対が書き込まれている。 |
| inode | ファイルに関するさまざまな情報。ファイルデータがある場合は、そのブロック番号など。touch(1)の例では、ファイルデータは無い。 |
| inodeビットマップ | ファイルシステム上のinode達が使用中・未使用かを表す。 |

# メタデータの書き込み順序

書き込み順序は参照関係に従うのが良い。
書き込まれていないものを参照しない状態を常に保てば、障害耐性の向上につながる。

参照関係

親ディレクトリブロック
↓
inode
↓
ファイルデータ

あるべき書き込み順序

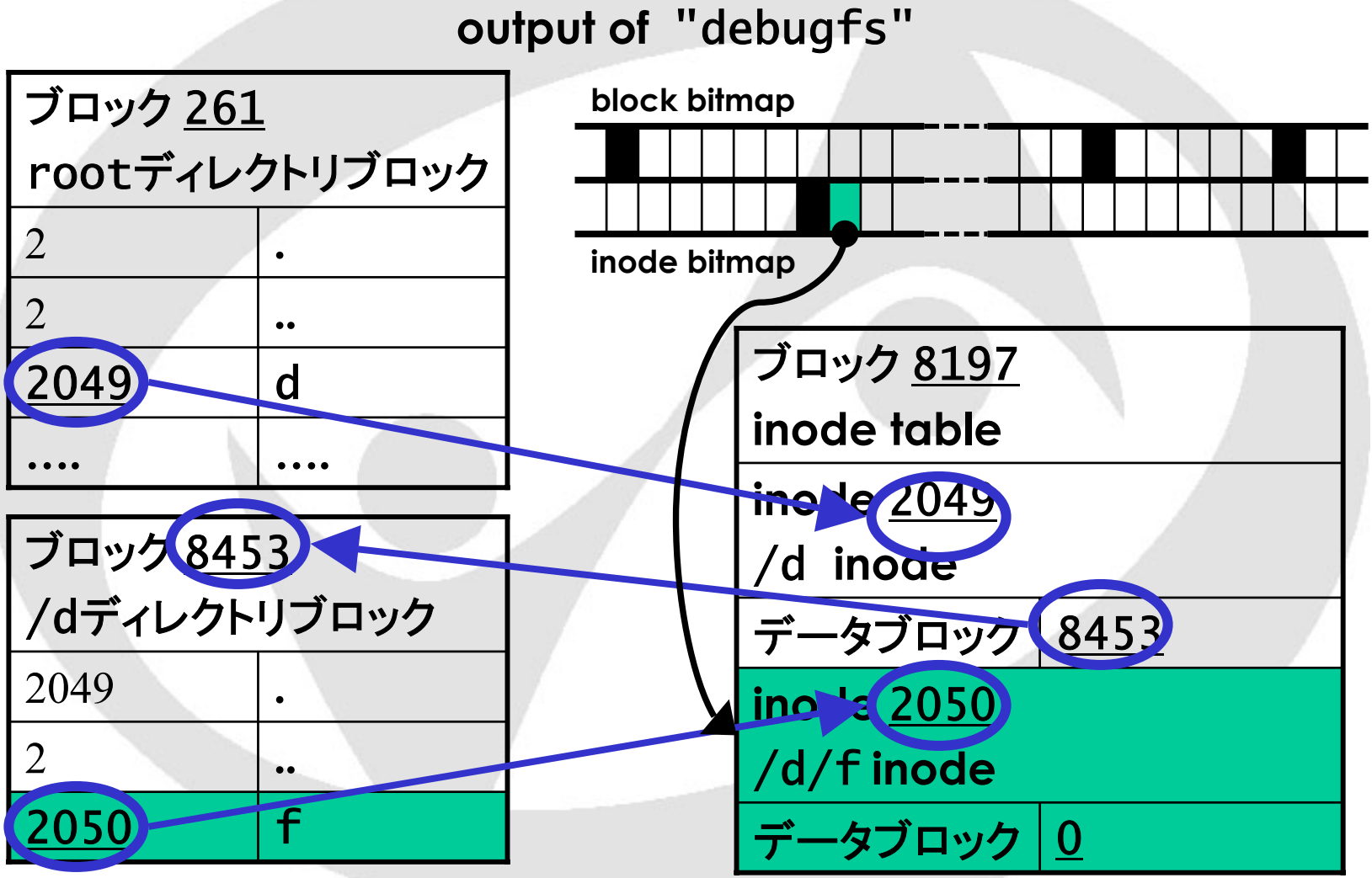
(ブロックビットマップ)
↓
(ファイルデータ)
↓
inodeビットマップ
↓
inode
↓
親ディレクトリブロック

# メタデータ更新の実例 creat(″/d/f″) -- 1

| Output of dumpe2fs | Block group 0 | Block group 1 |
|---|---|---|
| Superblock | 1(block number) | 8193 |
| G. descriptor | 2 | 8194 |
| Block bitmap | 3 | 8195 |
| inode bitmap | 4 | 8196 |
| inode table | 5 - 260 | 8197 - 8452 |

| Output of debugfs | / | /d | /d/f |
|---|---|---|---|
| inode | 2(inode number) | 2049 | 2050 |
| Type | Directory | Directory | Regular |
| Blocks | 261(block number) | 8453 | 8454 |

# メタデータの参照関係 creat("/d/f")--2



# 非同期書き出し

Systemcall発行時にdiskへの書き出しは行われず、unmount時、kernel threadのbdflushやkupdated により書き込まれる。

| | |
|---|---|
| bdflush | dirty buffer が一定以上たまったら動きだし、書き出しを行い、dirty buffer 数を一定数まで減らす |
| kupdated | 一定間隔をおいて起動し、一定時間を経過したdirty buffer を書き出す |

# 整合性とfsck

ファイルシステムの整合性
- kupdated, bdflushにより、バッファが書き出される前
- バッファが書き出された後
- バッファが書き出されている最中

予期せぬシステムダウン後のfsck
- 修復の方針
- 所要時間

# ext2 + jbd = ext3(1)

ファイルシステムの整合性

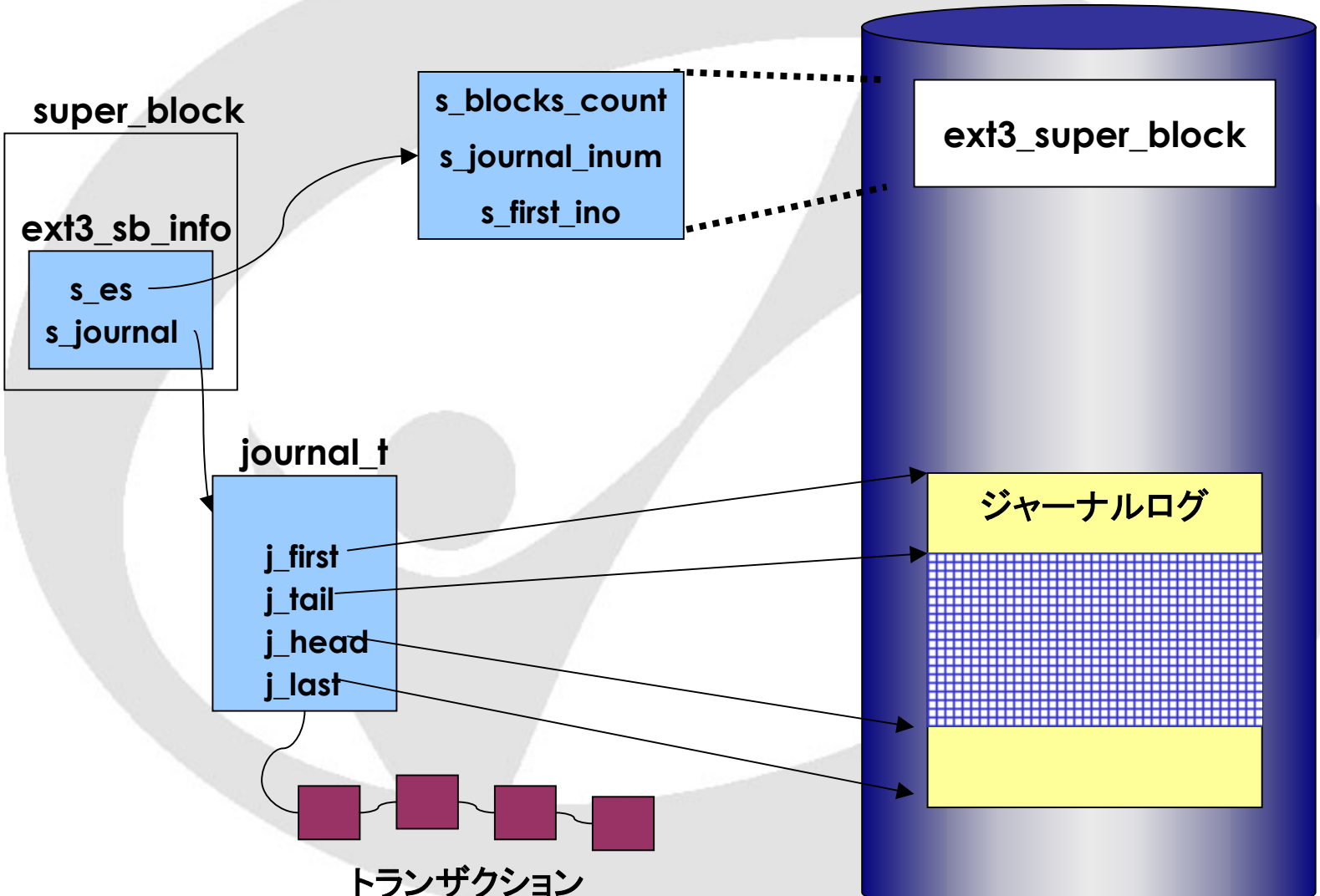
- 従来のkupdated, bdflushなどの動作を変えずに、メタデータなどの更新情報(ログ)を独立した領域に、五秒間隔で書き込む。

# ext2 + jbd = ext3(2)

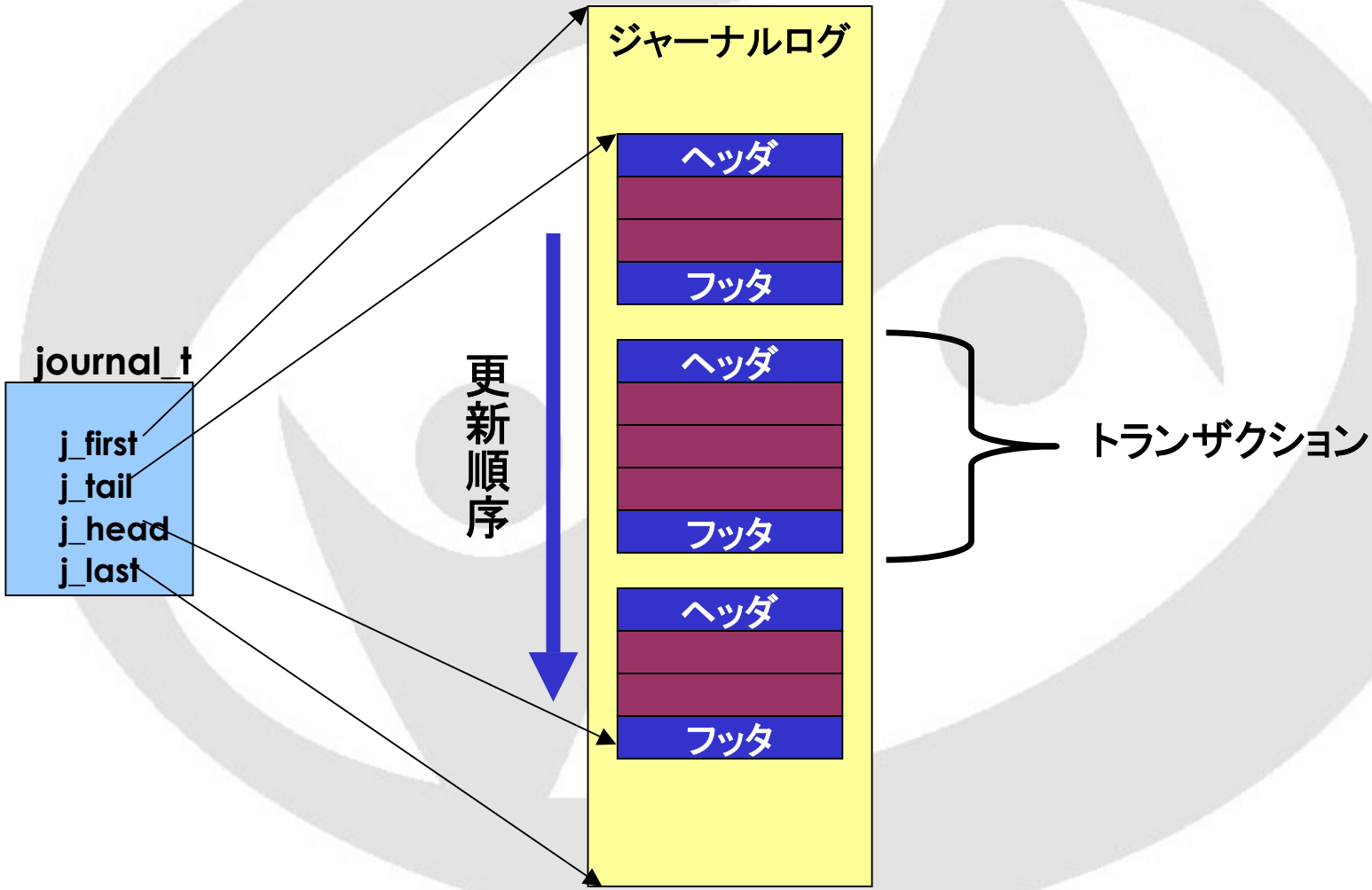
予期せぬシステムダウン後のfsck
- ジャーナルログからまるごと復旧できる。
- 対象範囲がジャーナルログだけなので、所要時間が非常に短い。

特徴
- ディスクレイアウトがext2fsと互換性がある。ジャーナル領域を予約inodeにつなぎ、ファイルデータとしている。
- jbdはファイルシステムに依存しておらず、汎用性と独立性が高い。

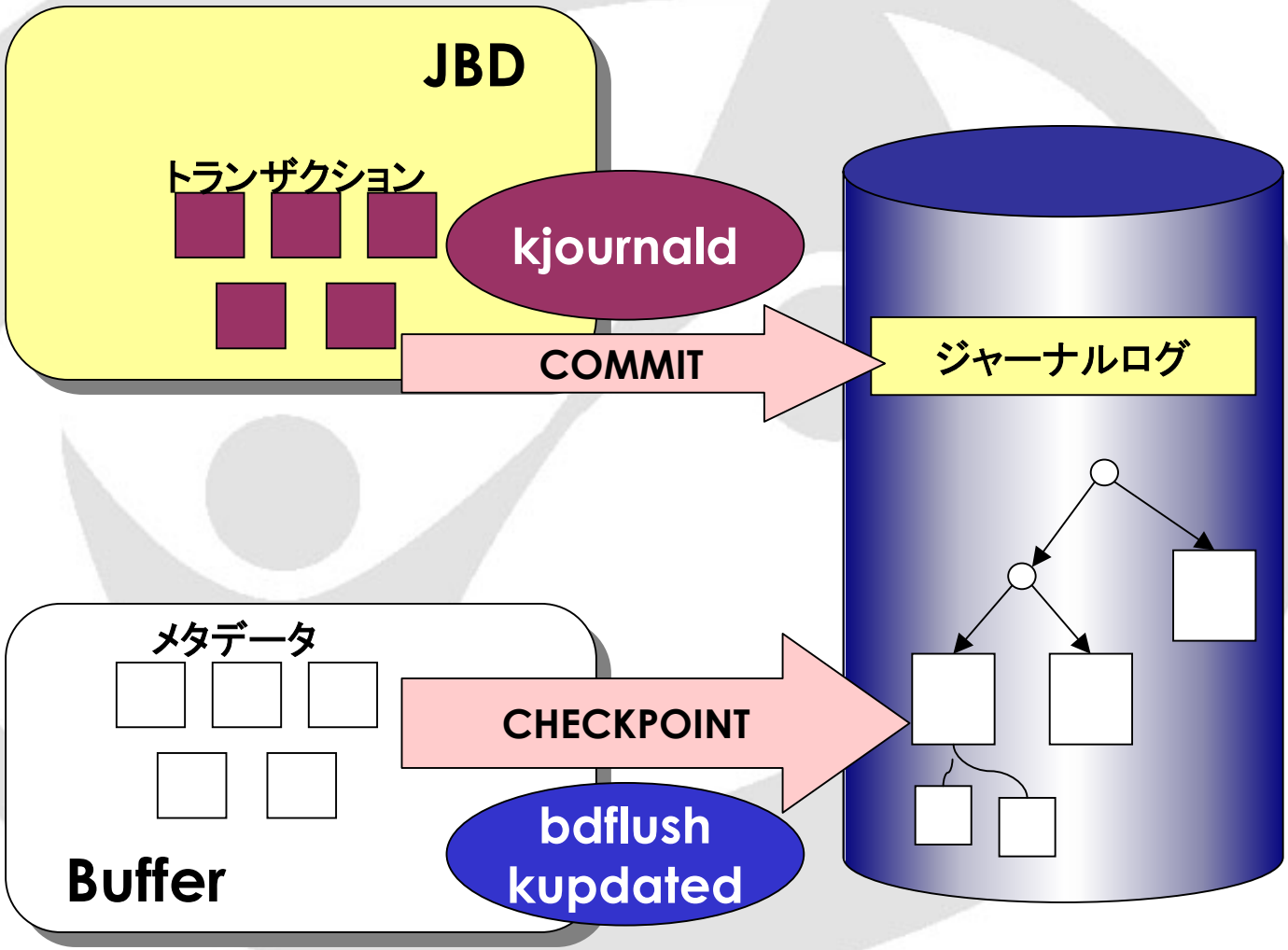
# EXT3ジャーナルデータ構造



# ジャーナルログ形式



# ジャーナルのCOMMITとCHECKPOINT



# ファイル生成時間を計る time(bash)

```
#!/bin/bash -
n=${1:-100}
while test $n -gt 0; do s="$s $n"; let n-=1; done
time touch $s
rm $s
```

| | ext3fs | | ext2fs | |
|---|---|---|---|---|
| | async | sync | async | sync |
| real | 0m0.364s | 0m9.364s | 0m0.055s | 0m2.184s |
| user | 0m0.010s | 0m0.010s | 0m0.000s | 0m0.000s |
| sys | 0m0.300s | 0m0.300s | 0m0.010s | 0m0.000s |

※非常に単純な測定！鵜呑みにするな。