



FUSE and OpenSolaris

Mark Phalan

Software Engineer

Sun Microsystems

Topics

- Background
- Architecture Overview
- FUSE Filesystems
- FUSE Messaging
- Mounting
- Design
- Current Status

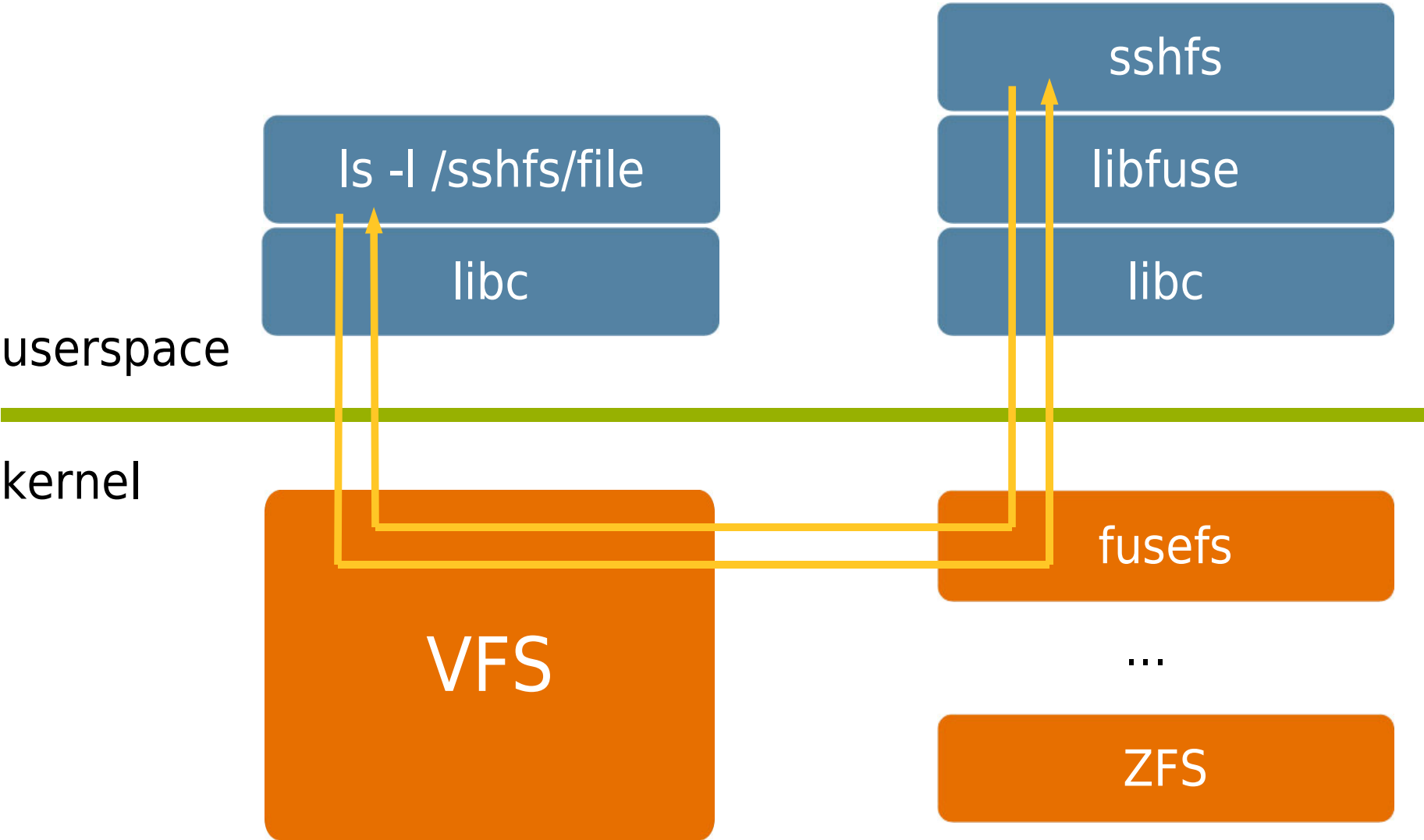
Background

- Developed on Linux
 - > 2.6.14+
- Later ported to FreeBSD and MacOS X
- NetBSD implementation
 - > Puffs and librefuse
- libfuse essentially unchanged for all Oses (except netbsd)
- OpenSolaris FUSE is a port of the FreeBSD implementation

Architecture Overview

- A framework which allows **F**ilesystems to be implemented in **U**Serspace
- Main components
 - > Kernel module
 - > Userspace library
 - > Mount utility (fusermount)
 - > FUSE filesystems
 - sshfs
 - ntfs-3g
 - ...

Architecture Overview...



Why Userspace Filesystems?

- Simpler to debug
 - > dbx
- Portable
- xBSD, MacOS X, Linux ...
- More secure
 - > Less privileged mounts
 - > Filesystem runs with user's privileges in userspace
- Easier to code
 - > Can make use of regular userspace libraries

Why Not Userspace Filesystems?

- Performance
 - > Context switching
 - > Movement of data
- Added complexity
 - > Admin model for user mounts
 - > More message passing

FUSE Filesystems

- Implement application specific functions for each necessary filesystem operation
 - > read
 - > write
 - > open
 - > close
 - > ...
- Interfaces defined by libfuse
- Invoke the entry point in libfuse
 - > Provide function pointers

FUSE Filesystems...

- libfuse reads from `/dev/fuse`
 - > Character device
 - > Blocking
 - > Waits for messages from the kernel
 - > Reads request, processes request and writes back a response
- Protocol/message formats specified in *`fuse_kernel.h`*
 - > Used by both library and kernel

FUSE Messaging

- Every request read from `/dev/fuse` has the following header

```
struct fuse_in_header {
    __u32 len;
    __u32 opcode;
    __u64 unique;
    __u64 nodeid;
    __u32 uid;
    __u32 gid;
    __u32 pid;
    __u32 padding;
};
```

- This is followed by an opcode specific payload structure

FUSE Messaging

- Every request written to /dev/fuse has the following header

```
struct fuse_out_header {  
    __u32 len;  
    __u32 error;  
    __u64 unique;  
};
```

- The unique identifier matches the unique identifier made in the request

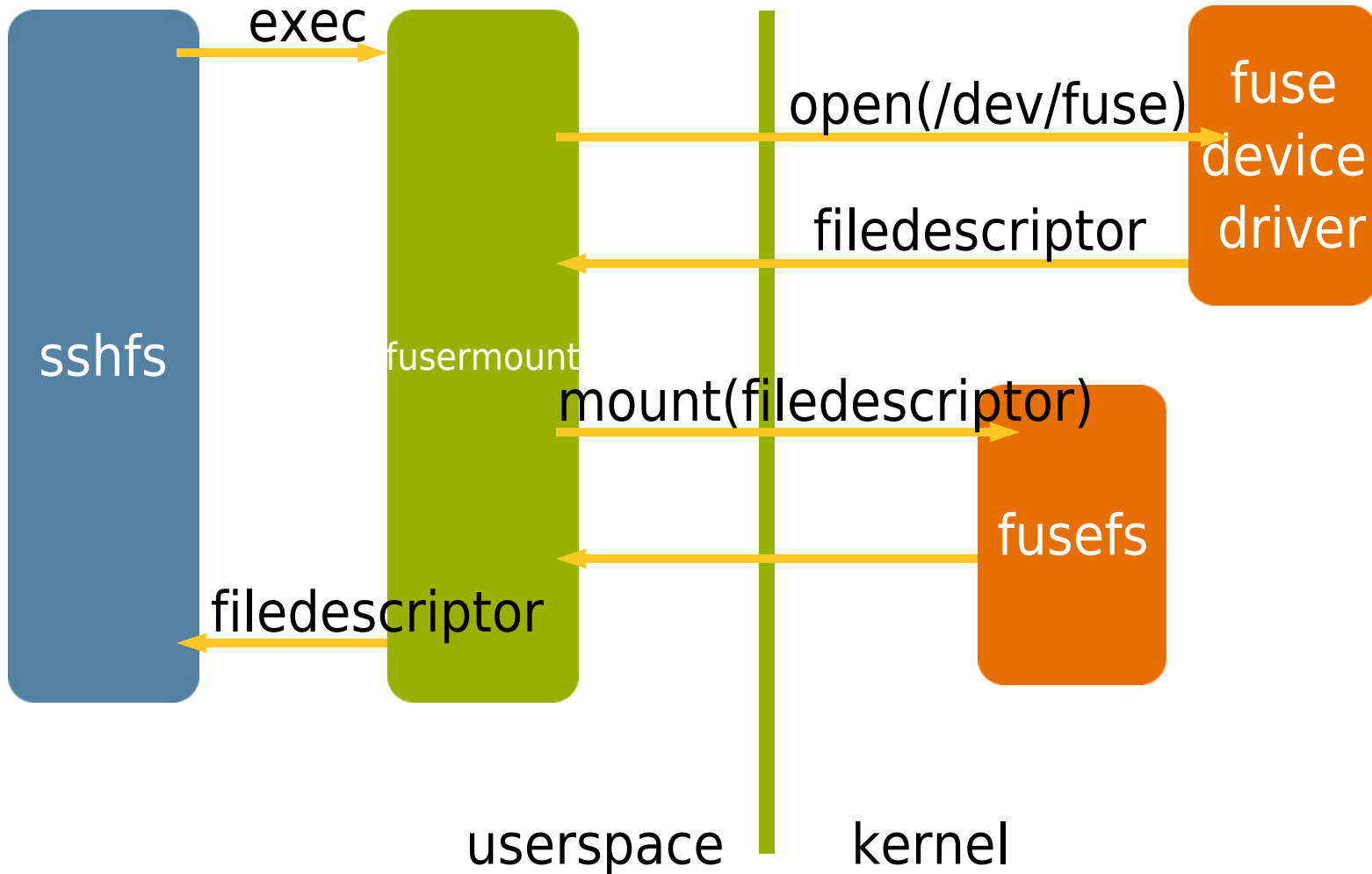
Mounting

- Somewhat complicated due to split nature of FUSE filesystems
 - > The userspace filesystem and the kernel module both need to take part
- User (less privileged) mounts are desirable
 - > Especially for non-block device mounts
 - > For e.g. sshfs
- Privileged (sys-mount) utility for mounting
 - fusermount

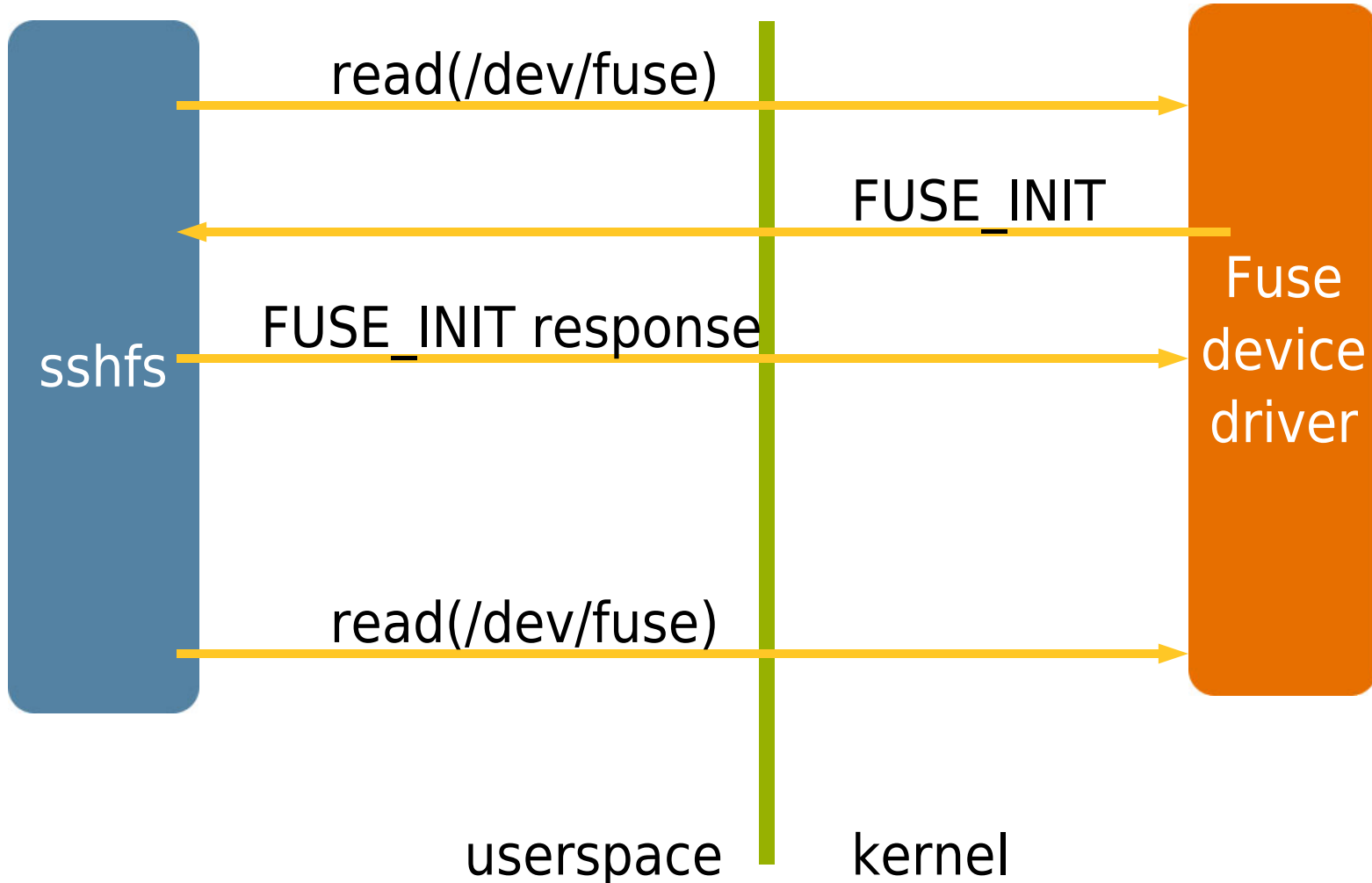
Mounting...

- Filesystem links against libfuse
 - > Runs with no extra privileges
- libfuse calls “fusermount” to do the actual mount(2)
 - > Small amount of code
 - > suid on Linux
 - > pfexec with sys-mount on OpenSolaris
- libfuse calls fusermount
 - > Performs the mount
 - > Gets out of the way

Mounting...



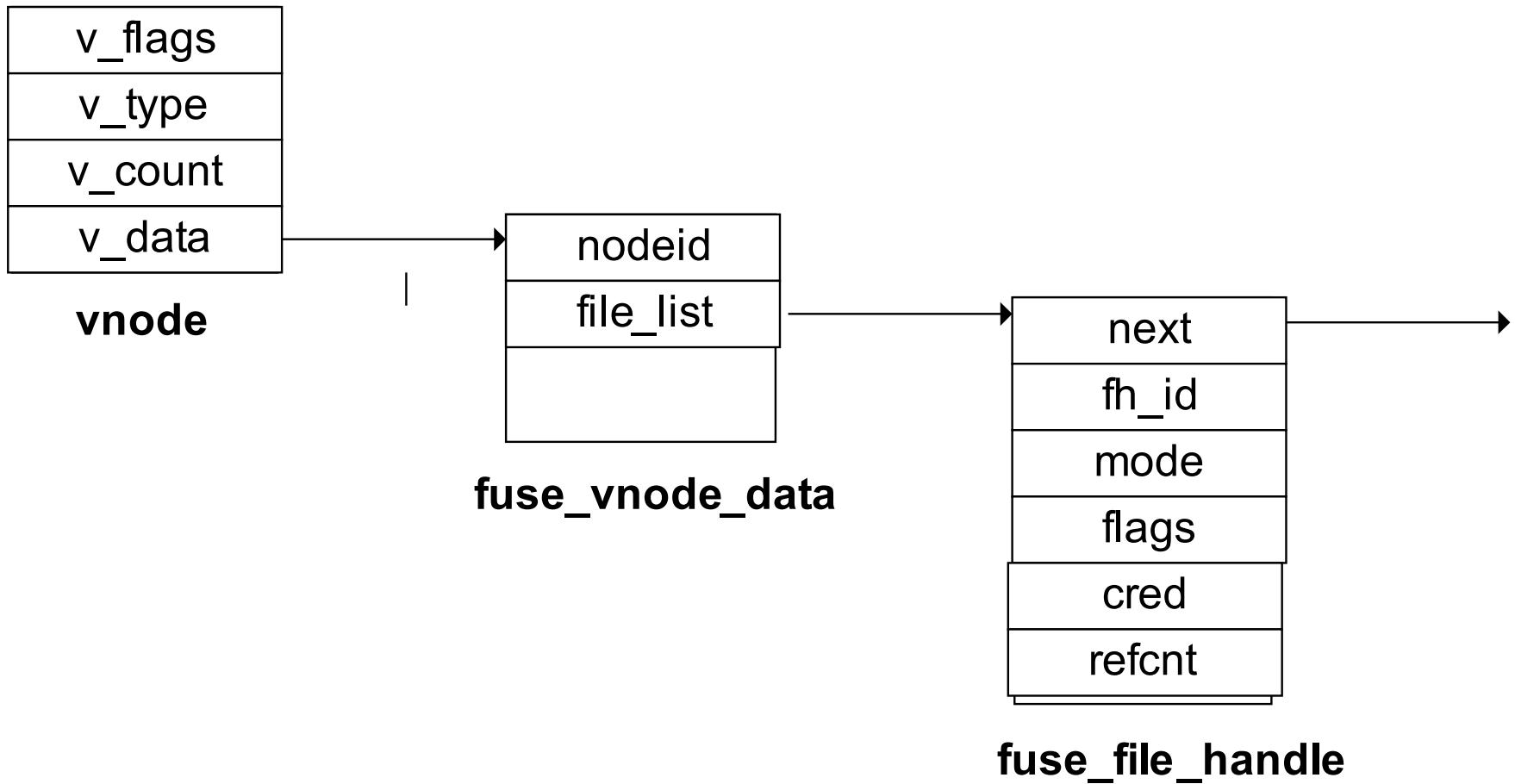
Mounting...



FUSE Filesystem Design...

- Solaris VFS is vnode based
- Every open file has a corresponding vnode
- There may be many file_t structures pointing to a single vnode
- Need to track all open files associated with each vnode
- vnodes stored in AVL tree
 - > Identified by nodeid

FUSE Filesystem Design...

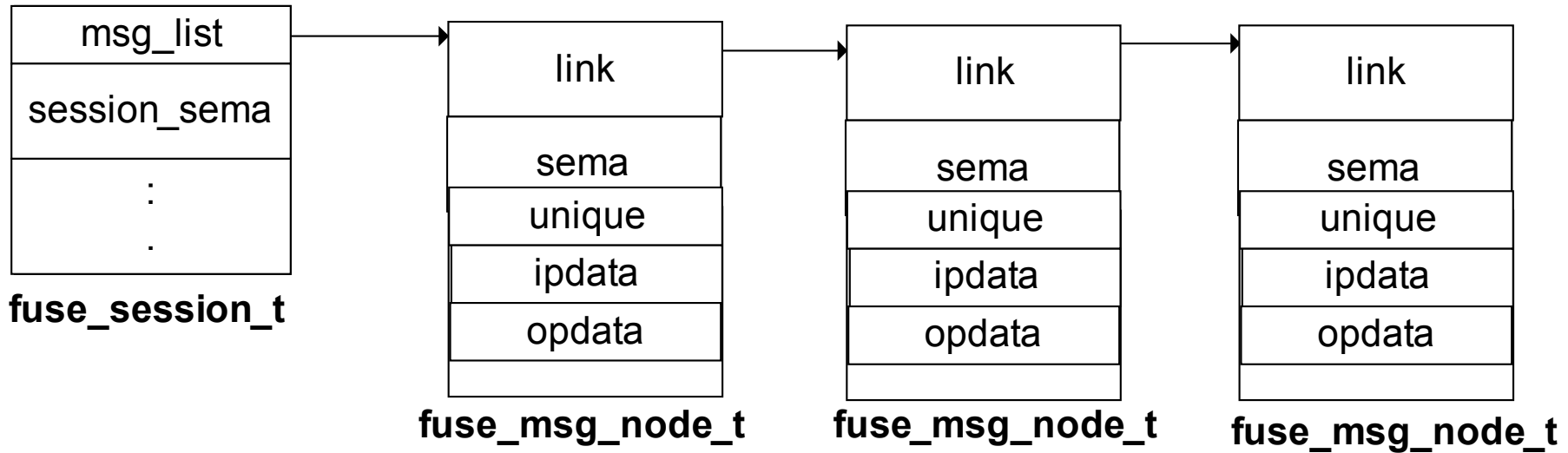


FUSE Filesystem Design...

- Each mounted fuse filesystem has a “session” structure

```
typedef struct fuse_session
{
    kmutex_t    session_mutex;
    ksema_t     session_sema; /* devops read sleeps over it */
    list_t      msg_list;    /* message awaiting service rest here */
    avl_tree_t  avl_cache; /* used to retrieve vnode from nodeid */
    uint32_t    minor;
    uint32_t    state;
    uint64_t    unique;      /* message identifier used between library and */
                                /* kernel module */
    cred_t      *usercred;   /* Credentials passed by fuse library */
    uint32_t    max_write;   /* Max Write value passed by fuse library */
                                /* during mount */
} fuse_session_t;
```

FUSE Filesystem Design...



- Requests inserted at the tail of the list
 - > e.g. `open(file)`
- Reads from `/dev/fuse` taken from the head

FUSE Filesystem Design...

- Every message is uniquely identified
 - > Minor number
 - > Unique 64 bit number (from session)
- Header and payload
- Queued in a per-session list

How it Started

- Two independent beginnings
 - > SEED Program
 - > MDE
- Current OpenSolaris implementation based on MDE work
- Internal demand for
 - > davfs2, encfs
- Nice to have
 - > ntfs-3g, sshfs, ...

Current status

- Still (basically) unfunded
- “Spare” time project
 - > Sarah Jelinek
 - > Mark Phalan
- Part-time project work
 - > Sunil Kumar
- May be changing...

Another Implementation

- Student project at MFF
 - > Tomas Tuma
 - > Jiri Tlach
 - > David Senkerik
- Complete re-implementation
- Read-only
- Not integrated into our code (yet)

Current Status

- All vnops implemented
- What still needs to be done
 - > Locking
 - > mmap()
 - > Performance
 - > Dtrace() probes
 - > Bugs
 - > Testing
- PSARC review
- Integration

Demo



Thanks!

Mark Phalan

mark.phalan@sun.com