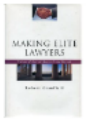
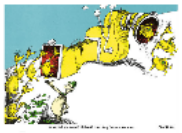
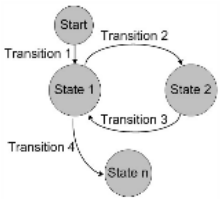
## A Composite IDL

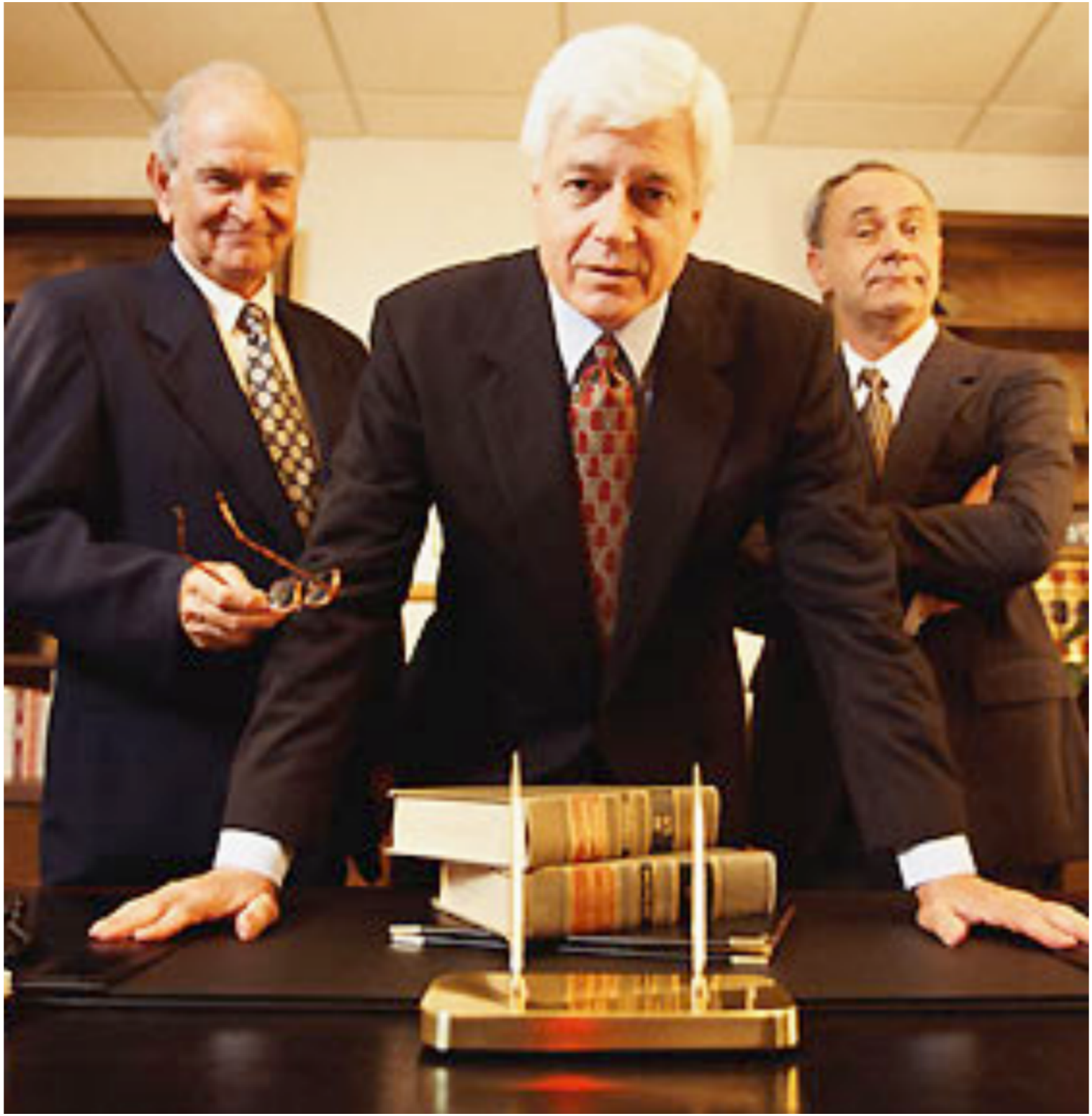


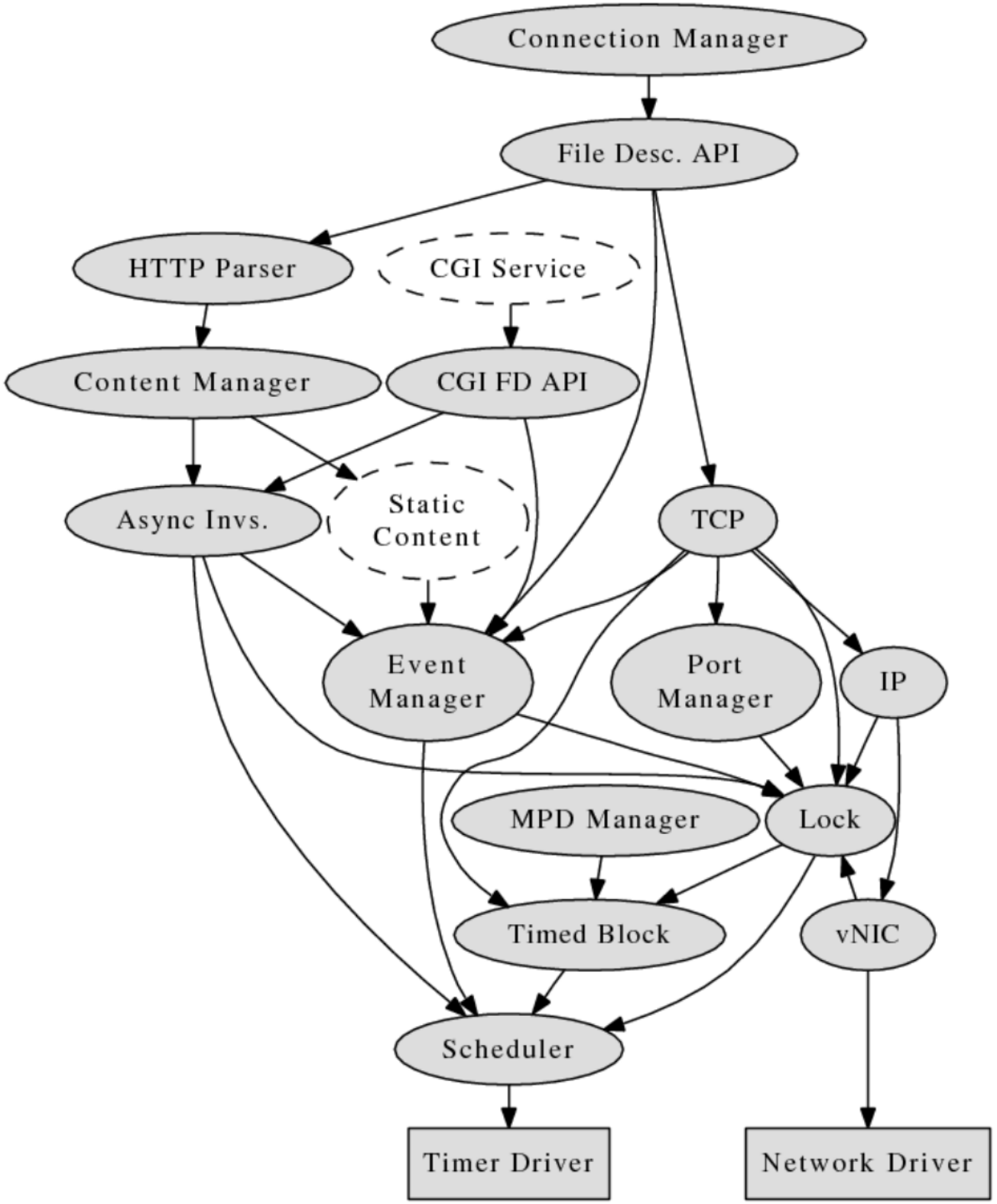





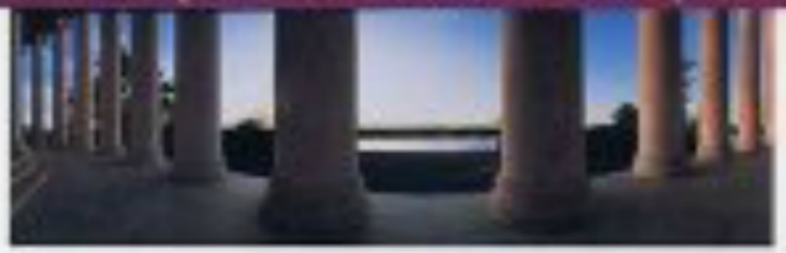

# A Composite IDL




Connection Manager
File Desc. API
HTTP Parser
CGI Service
Content Manager
CGI FD API
Async Invs.
Static Content
TCP
Event Manager
Port Manager
IP
MPD Manager
Lock
Timed Block
vNIC
Scheduler
Timer Driver
Network Driver

# MAKING ELITE LAWYERS

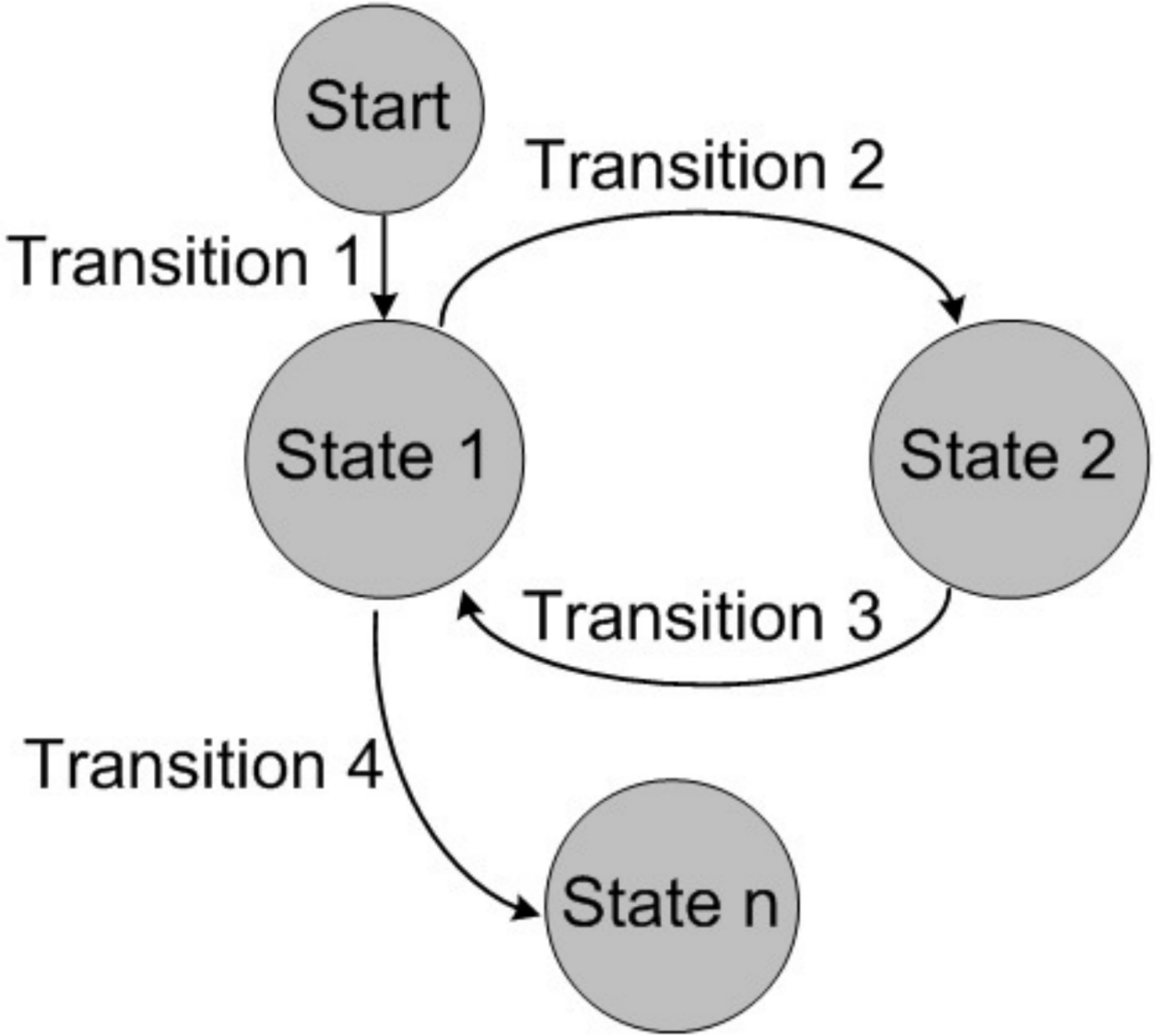
*Visions of Law at Harvard and Beyond*



Robert Granfield


SYLVESTER McMONKEY McBEAN'S STAR-BELLY SNIFFETCH MACHINE

DR. SEUSS


Start
Transition 2
Transition 1
State 1
State 2
Transition 3
Transition 4
State n

/bin/bash

/bin/bash 157x43

```
andrew@Rothbard ~/SchoolNotes/compsci/senior_design/composite-langs/Language/Composite/IDL $ ./CStub test.pony.c
typedef int spdid_t;
typedef int td_t;
typedef int tor_flags_t;
struct __sg_tsplit_data {
  td_t tid;
  tor_flags_t tflags;
  long evtid;
  char data[0];
};
td_t tsplit_call(spdid_t spdid, td_t tid, char * param, int len, tor_flags_t tflags, long evtid, char * param2, int len2) {
  long fault = 0;
  td_t ret;
  struct __sg_tsplit_data *d;
  cbuf cb;
  unsigned int sz = len + len2 + sizeof(struct __sg_tsplit_data);
  assert(param && len >= 0);
  assert(param[len] == '\0');
  assert(param2 && len2 >= 0);
  assert(param2[len2] == '\0');
  d = cbuf_alloc(sz,&cb);
  if (!d) return -6;
  d -> tid = tid;
  d -> tflags = tflags;
  d -> evtid = evtid;
  memcpy(&d -> data[0],param,len);
  memcpy(&d -> data[0] + len,param2,len2);
  CSTUB_ASM_3(tsplit,spdid,cb,sz);
}
struct __sg_tsplit_data {
  td_t tid;
  tor_flags_t tflags;
  long evtid;
  char data[0];
};
td_t tsplit_call(spdid_t spdid, td_t tid, char * param, int len, tor_flags_t tflags, long evtid, char * param2, int len2) {
  long fault = 0;
  td_t ret;
  struct __sg_tsplit_data *d;
  cbuf cb;
  unsigned int sz = len + len2 + sizeof(struct __sg_tsplit_data);
  assert(param && len >= 0);
  assert(param[len] == '\0');
```

Menu /bin/bash 46 °F Thu Feb 14, 6:28 PM

Prezi

emacs@Rothbard


emacs@Rothbard

File Edit Options Buffers Tools Haskell nXhtml Help

```haskell
createStubCode :: String -> Fix Sem -> [Field] -> [Fix Sem]
createStubCode fnname rtype fs =
  let
    params = arguments' $ map (\(t, n) -> variable' t (name' n) nil') fs
    fnname' = fnname ++ "_call"
    structType = createStubStructName fnname
    intZero = cint' 0
    intRet = cint' (-6)
    (spdid, strLengs, fields) = getFields fs
    asserts = stringAsserts strLengs
    placements = placeFields fields
    instructions1 = [
                      variable' (int' signed' []) (name' "fault") intZero
                    , variable' rtype (name' "ret") nil'
                    , variable' (pointer_to' structType) (name' "d") nil'
                    , variable' (builtin' (name' "cbuf_t")) (name' "cb") nil'
                    , lengthInstruction fnname strLengs
                    ]
    cbPtr = unary' (name' "&") (name' "cb")
    allocCall = funcall' (name' "cbuf_alloc") [name' "sz", cbPtr]
    instructions2 = [
                      binary' (name' "d") (name' "=") allocCall
                    , ifthen' (unary' (name' "!") (name' "d")) (return' intRet)
                    ]
    spdList = case spdid of
      Nothing  -> []
      Just spd -> [name' spd]
    asmParams = [name' fnname] ++ spdList ++ [name' "cb", name' "sz"]
    asm = funcall' (name' $ "CSTUB_ASM_" ++ (show $ (length asmParams) - 1)) asmParams
    strings = (fillLengthArray strLengs) ++ (placeStrings strLengs)
    instructions3 = [
                      asm
                    , funcall' (name' "cbuf_free") [name' "d"]
                    , return' (name' "ret")
                    ]
    instructions = instructions1 ++ asserts ++ instructions2
                   ++ placements ++ strings ++ instructions3
    lenLength = fromIntegral $ length strLengs + 1
    lenAry = array' (int' signed' []) (cint' lenLength)
    fields' = fields ++ [(lenAry, "len")]
```

--:---  CStub.hs  6% L11  Git:master  (Haskell Ind Flymake:1/0 Doc)