Abstraction for Programmming

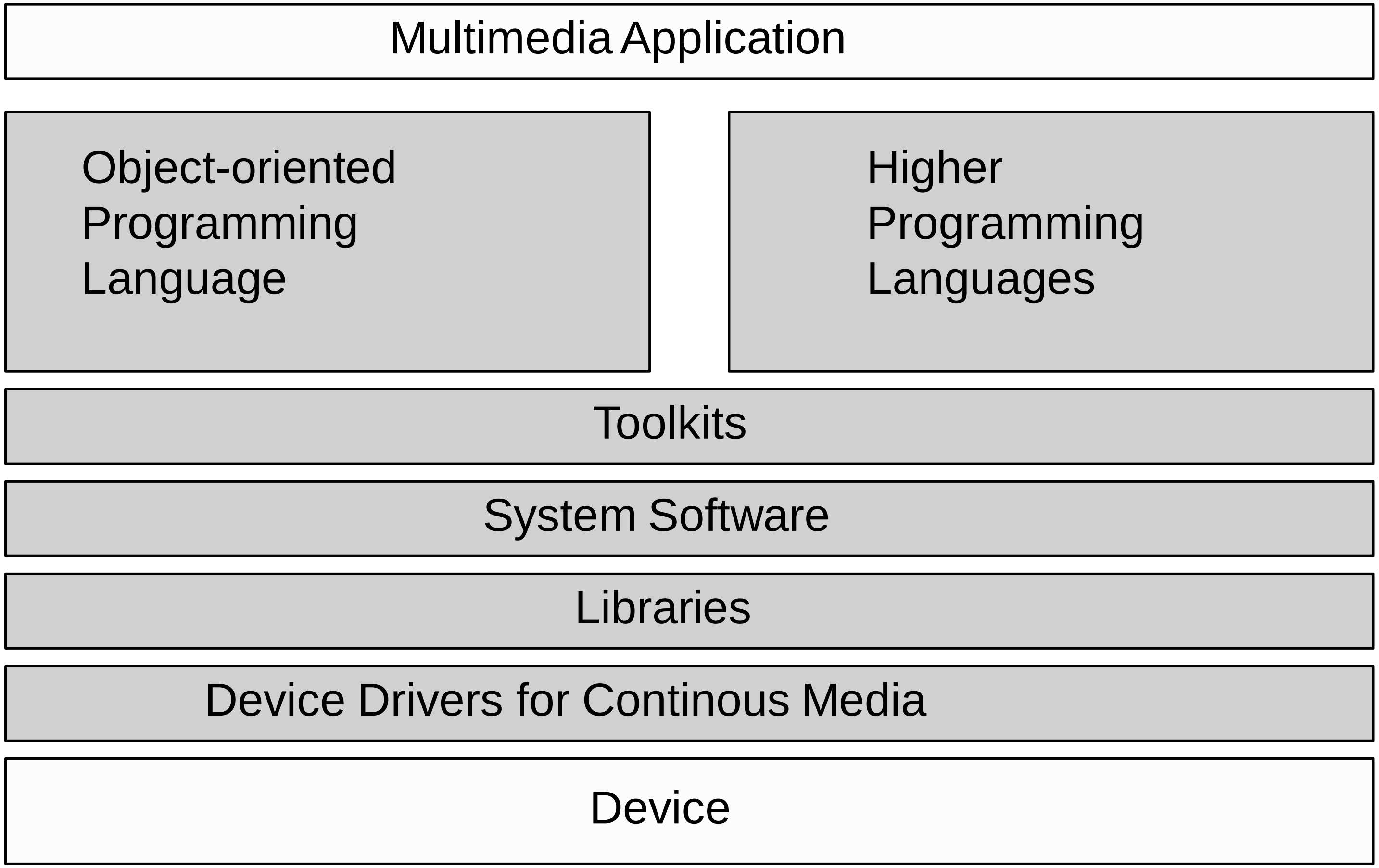
The state of the art of programming

- Most of the current commercially available multimedia applications are implemented in procedure-oriented programming languages.
- Application code is still highly dependent on hardware.
- Change of multimedia devices still often requires re-implementation.
- Common operating system extensions try to attack these problems
- Different programming possibilities for accessing and representing multimedia data

Overview of different abstraction levels

- Libraries
- System Software
- Toolkits
- Higher Programming Languages
- Object-oriented Approaches

14.2 Abstraction Levels



Multimedia Application

Object-oriented Programming Language

Higher Programming Languages

Toolkits

System Software

Libraries

Device Drivers for Continous Media

Device

14.3 Libraries

Processing of continuous media based on functions embedded into libraries

Libraries differ in their degree of abstraction

Example from IBM's early Audio Visual Connection (AVC):

```
acb.channel = AAPI_CHNA

acb.mode = AAPI_PLAY

...

aud_init(&acb)  /* acb is the audio control block */

...

audrc = fab_open(AudioFullFileName,AAFB_OPEN,AAFB_EXNO, 0,&fab,0,0,0,0);

fork(START in PARALLEL)

aud_strt(&acb)

displayPosition(RelativeStarttime, Duration)

...
```

14.4 System Software

Device access becomes part of the operating system:

Data as Time Capsules (file extensions)

- each Logical Data Unit (LDU) carries in its time capsule its data type, actual value and valid life span
- useful concept for video, where each frame has a valid life span of 40ms (rate of read access during a normal presentation)
- presentation rate is changed for VCR (Video Casette Recorder) functions like fast forward, slow forward or fast rewind by
  - changing the presentation life span of a LDU
  - skipping of LDUs or repetition of LDUs

Data as Streams

- a stream denotes the continuous flow of audio and video data between a source and a sink
- prior to the flow the stream is established equivalent to the setup of a connection in a networked environment

14.5 Toolkits

Simpler approach than the system software interface from the users point of view are Toolkits (simpler because abstarction from many “uninteresting“ details) :

- abstract from the actual physical layer
- allow a uniform interface for communication with all different devices of continuous media
- introduce the client-server paradigm
- can hide the process-structures
- can be embedded into programming languages or object-oriented environments

14.6 Higher Programming Languages

Media as Types:

- definition of appropriate data types (e.g. for video and audio)
- smallest unit can be a LDU
- example of merging a text and a motion picture (OCCAM-2 similar notation):

```
subtitle  TEXT_STRING
mixed.video, ldu.video VIDEO_LDU;
...
WHILE
 COBEGIN
  PROCESS_1
   input(av_filehandle,ldu.video)
   IF new_video_scene
    input(subtitle_filehandle,subtitle)
   mixed.video := ldu.video + subtitle
```

```
PROCESS_2
  output(video_window,mixed.video)
...
END_WHILE
...
```

In above example implicit type conversion must occur

14.6 Higher Programming Languages

Media as Files:

- instead of considering continuous media as data types they can be considered as files

```
file_h1 = open(MICROPHONE_1,...)
file_h2 = open(MICROPHONE_2,...)
file_h3 = open(SPEAKER, ...)
...
read(file_h1)
read(file_h2)
mix(file_h3, file_h1, file_h2)
activate(file_h1, file_h2, file_h3)
...
deactivate(file_h1, file_h2, file_h3)
...
rc1 = close(file_h1)
rc2 = close(file_h2)
rc3 = close(file_h3)
```

14.6 Higher Programming Languages

Media as Processes:

- it is possible to map continous media to processes and integrate them into an HLL

```
PROCESS cont_process_a;
...
On_message_do
  set_volume ...
  set_loudness ...
  ...
...
[main]
pid = create(cont_process_a)
send(pid, set_volume, 3)
send(pid, set_loudness)
...
```

This process implements a set of actions (“`set-volume`“, “`set-loudness`“)

14.7 Programming Language Requirements

The High Level Language (HLL) should support parallel processing, because the processing of continuous data is

- controlled by the HLL through pure asynchronous instructions
- an integral part of a program through the identification of media

Different processes must be able to communicate through an Inter-Process-Communication mechanism (IPC), which must be able to:

- understand a priori and/or implicitly specified time requirements (QoS parameters or extracted from the data type)
- transmit the continous data according to the requirements
- initiate the processing of the received continous process on time

14.8 Object-oriented Approaches

Basic ideas of object-oriented programming are data encapsulation inheritance, in connection with class and object definitions.

- Abstract Type Definition (definition of data types through abstract interfaces)
- Class (implementation of a abstract data type)
- Object (instance of a class)

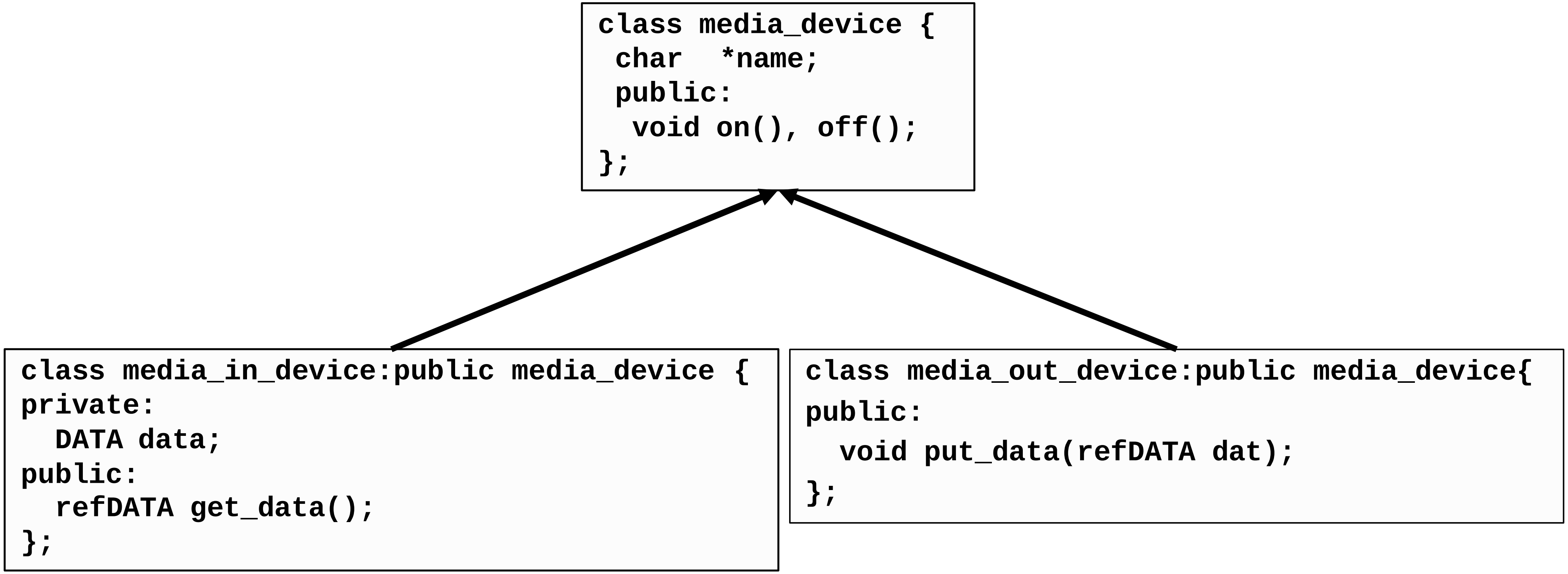
Other important properties of object-oriented systems are:

- Inheritance
- Polymorphism

14.8 Object-oriented Approaches

Devices as Classes:

- devices are assigned to objects which represent their behaviour and interface



```
class media_device {
 char  *name;
 public:
  void on(), off();
};
```

```
class media_in_device:public media_device {
private:
  DATA data;
public:
  refDATA get_data();
};
```

```
class media_out_device:public media_device{
public:
  void put_data(refDATA dat);
};
```

14.8 Object-oriented Approaches

Processing Units as Classes

Three main objects:

- source objects
- destination objects
- combined source-destination objects allows the creation of data flow paths through connection of objects

Multimedia Object

- Basic Multimedia Classes (BMCs) /
  Basic Multimedia Objects (BMOs)
- Compound Multimedia Classes (CMCs) /
  Compound Multimedia Objects (CMO), which are compound of BMCs / BMOs and other CMCs/CMOs
- BMOs and CMOs can be distributed over different computer nodes

14.8 Object-oriented Approaches

Media as Classes:

- Media Class Hierarchies define hierarchical relations for different media
- different class hierarchies are better suited for different applications