VERIFICATION ACADEMY

Basic OVM

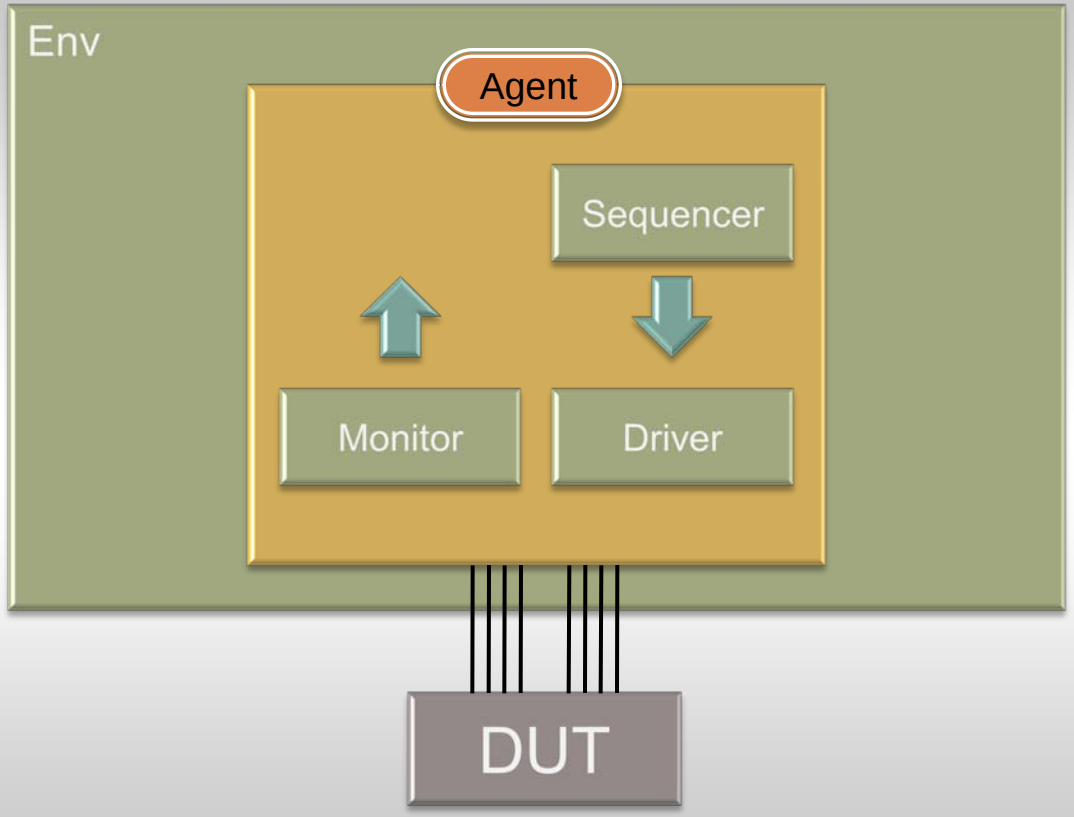
Connecting Components

John Aynsley
CTO, Doulos

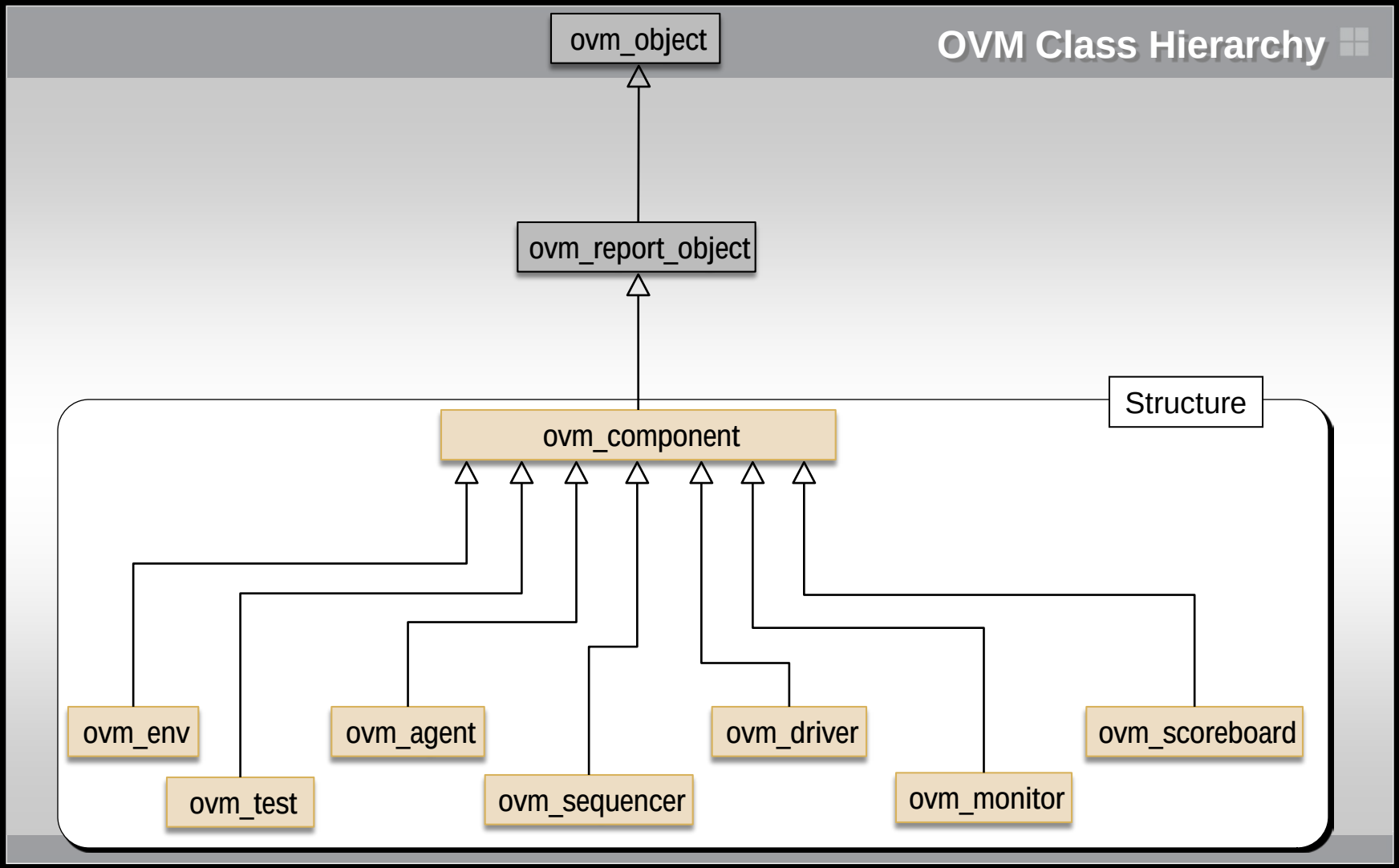
academy@mentor.com
www.verificationacademy.com

Mentor Graphics®

Verification Components



OVM Class Hierarchy



Component

```
class my_agent extends ovm_agent;
```

Component

```
class my_agent extends ovm_agent;
  `ovm_component_utils(my_agent)

  my_sequencer my_sequencer_h;
  my_driver    my_driver_h;
```

Component

```
class my_agent extends ovm_agent;
  `ovm_component_utils(my_agent)

  my_sequencer my_sequencer_h;
  my_driver    my_driver_h;

  function new(string name, ovm_component parent);
    super.new(name, parent);
  endfunction: new

  function void build;
    ...

  function void connect;
    ...
```

Factory Method

```
function void build;
  super.build();
  my_sequencer_h =
    my_sequencer::type_id::create("my_sequencer_h", this);

  my_driver_h    =
    my_driver   ::type_id::create("my_driver_h"   , this);
endfunction: build
```

Instance name

Parent

"Factory methods" can be overridden in tests

Connect

```
function void build;
  super.build();
  my_sequencer_h =
    my_sequencer::type_id::create("my_sequencer_h", this);

  my_driver_h    =
    my_driver   ::type_id::create("my_driver_h"   , this);
endfunction: build

function void connect;
  my_driver_h.seq_item_port.connect(
    my_sequencer_h.seq_item_export );
endfunction: connect
```

Connect

```
function void build;
  super.build();
  my_sequencer_h =
    my_sequencer::type_id::create("my_sequencer_h", this);

  my_driver_h    =
    my_driver   ::type_id::create("my_driver_h"   , this);
endfunction: build

function void connect;
  my_driver_h.seq_item_port.connect(
    my_sequencer_h.seq_item_export );
endfunction: connect
```

Phase Methods

```
class my_component extends ovm_component;
  ...
```

Phase Methods

```
class my_component extends ovm_component;
  ...
  function new(string name, ovm_component parent);
    super.new(name, parent);
  endfunction: new
```

Phase Methods

```
class my_component extends ovm_component;
  ...
  function new(string name, ovm_component parent);
    super.new(name, parent);
  endfunction: new
  function void build;
    super.build();
    ...
```

Called top-down

Phase Methods

```
class my_component extends ovm_component;
  ...
  function new(string name, ovm_component parent);
    super.new(name, parent);
  endfunction: new
  function void build;
    super.build();
    ...
  function void connect;
    ...
```

Called bottom-up

Phase Methods

```
class my_component extends ovm_component;
  ...
  function new(string name, ovm_component parent);
    super.new(name, parent);
  endfunction: new
  function void build;
    super.build();
    ...
  function void connect;
    ...
  function void start_of_simulation;
    ...
```

Phase Methods

```
class my_component extends ovm_component;
  ...
  function new(string name, ovm_component parent);
    super.new(name, parent);
  endfunction: new
  function void build;
    super.build();
    ...
  function void connect;
    ...
  function void start_of_simulation;
    ...
  task run;
    ...
```

The only task

Phase Methods

```
class my_component extends ovm_component;
  ...
  function new(string name, ovm_component parent);
    super.new(name, parent);
  endfunction: new
  function void build;
    super.build();
    ...
  function void connect;
    ...
  function void start_of_simulation;
    ...
  task run;
    ...
  function report;
    ...
```

Anatomy of an OVM Component

```
class my_component extends ovm_component;
  `ovm_component_utils(my_component)

  virtual dut_if dut_if_h;

  my_sequencer my_sequencer_h;
  my_driver    my_driver_h;

  function new(string name, ovm_component parent); ...
  function void build; ...
  function void connect; ...
  function void start_of_simulation; ...
  task run; ...
  function check; ...
  function report; ...
endclass
```

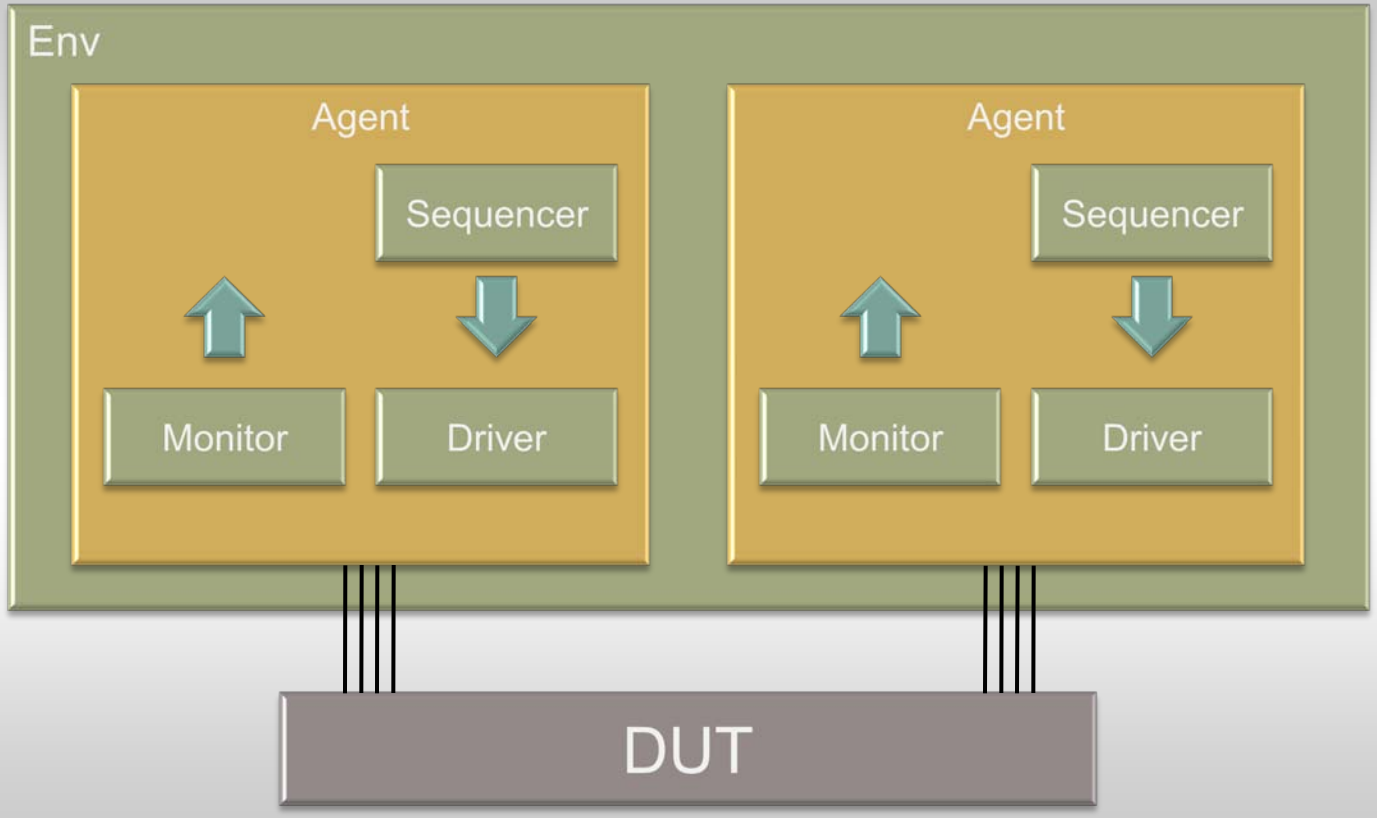
Factory registration

External interfaces

Internal component handles

Standard phase methods

Summary


Env
Agent
Sequencer
Monitor
Driver
Agent
Sequencer
Monitor
Driver
DUT

VERIFICATION ACADEMY

Basic OVM

Connecting Components

John Aynsley
CTO, Doulos

academy@mentor.com
www.verificationacademy.com

Mentor Graphics®