Order Management Capability

Automated order processing is hard

Traditional implementation with conditions leads to a complete mess

```
if…else        if…
if…else        else if…
if…else…       else…
```

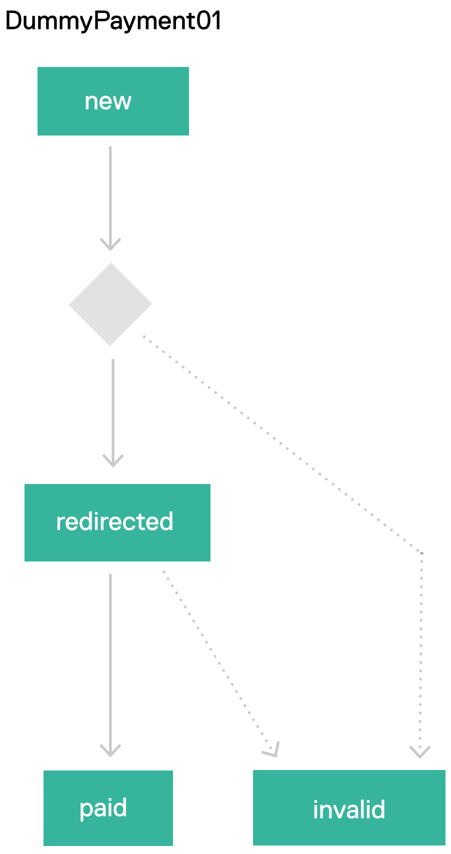
Spryker utilizes state machines

Spryker uses state machines to modelize, visualize and execute the order management system (OMS). This allows up to an 100% automated processes (even without an ERP system) and enables cross-department communication in companies.

States

The lifecycle of an order is presented in states.

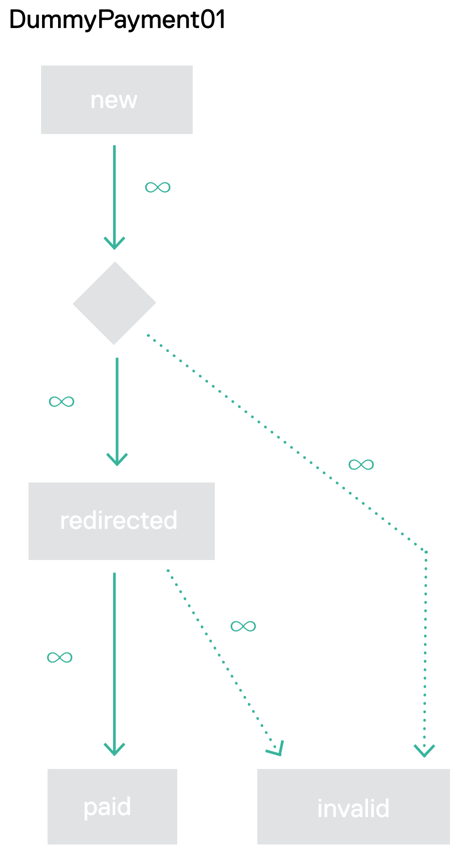
```
<states>
    <state name="new"/>
    <state name="redirected"/>
    <state name="paid"/>
    <state name="invalid"/>
</states>
```

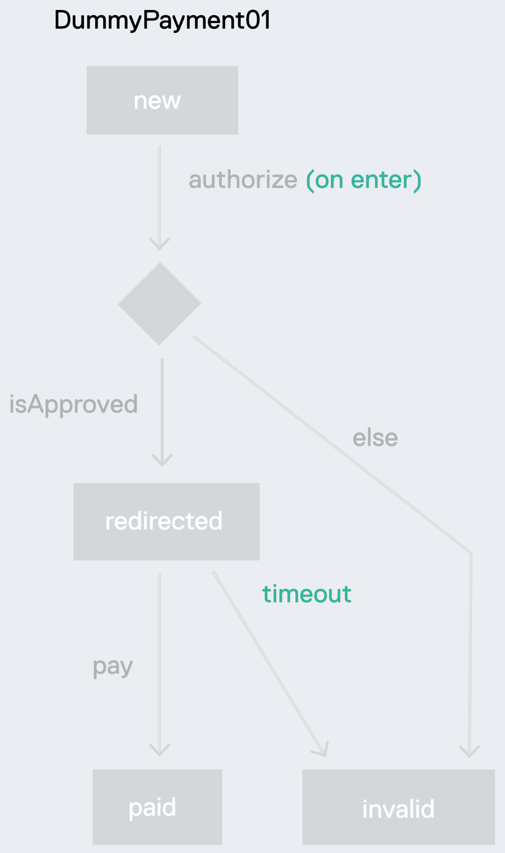


Transitions

Transitions are directed connections between different states, one transition connects two states together.

```
<transitions>

    <transition happy="true">
        <source>new</source>
        <target>redirected</target>
    </transition>

    <transition>
        <source>new</source>
        <target>invalid</target>
    </transition>

    <transition happy="true">
        <source>redirected</source>
        <target>paid</target>
    </transition>

    <transition>
        <source>redirected</source>
        <target>invalid</target>
    </transition>

</transitions>
```





Events

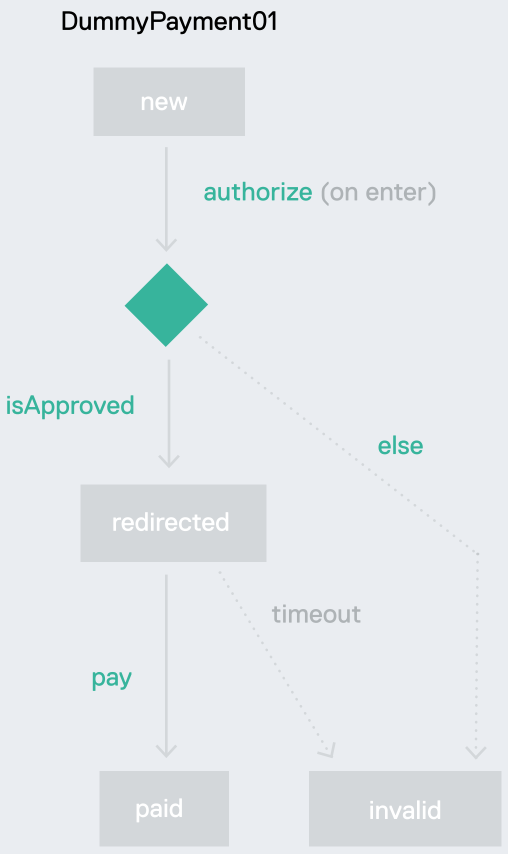
Events are triggered to run a transition from the source state to the target one. To have full control of the order process, there are four kind of events: automatic, manual, timeout, and external API calls.

```
<transitions>

    <transition happy="true">
        <source>new</source>
        <target>redirected</target>
        <event>authorize</event>
    </transition>

    <transition>
        <source>new</source>
        <target>invalid</target>
        <event>authorize</event>
    </transition>

    <transition happy="true">
        <source>redirected</source>
        <target>paid</target>
        <event>pay</event>
    </transition>

    <transition>
        <source>redirected</source>
        <target>invalid</target>
        <event>timeout</event>
    </transition>

</transitions>
<events>
    <event name="authorize" onEnter="true"/>
    <event name="pay"/>
    <event name="timeout"/>
</events>
```

Conditions

Conditions are the replacement of the if-else conditions in a traditional process implementation.

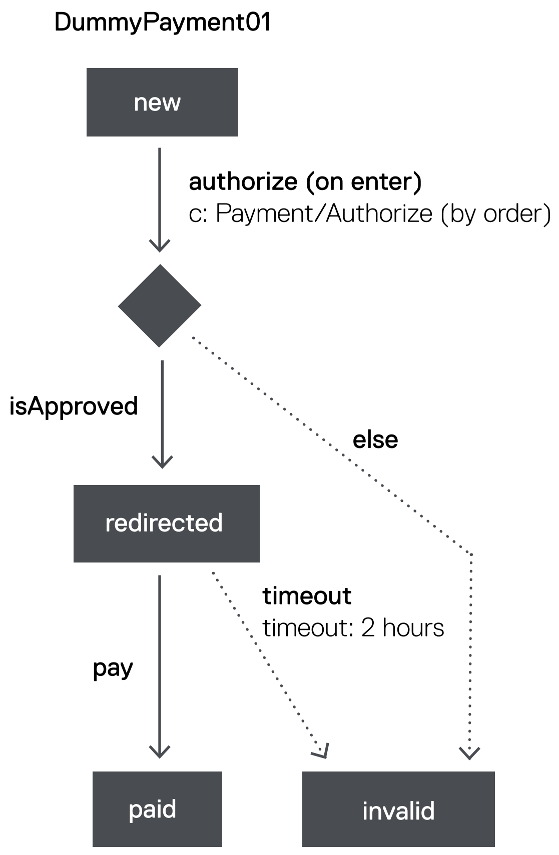
```
  <transition happy="true" condition="isApproved">
    <source>new</source>
    <target>redirected</target>
    <event>authorize</event>
</transition>

<transition condition="isError">
    <source>new</source>
    <target>invalid</target>
    <event>authorize</event>
</transition>
```



Commands

Commands are the extra functionalities added to an event, e.g. sending an email while moving from the source state to the target one.

```
<events>
    <event name="authorize"
           onEnter="true"
           command="Payment/Authorize"/>
    <event name="pay"/>
    <event name="timeout"/>
</events>
```

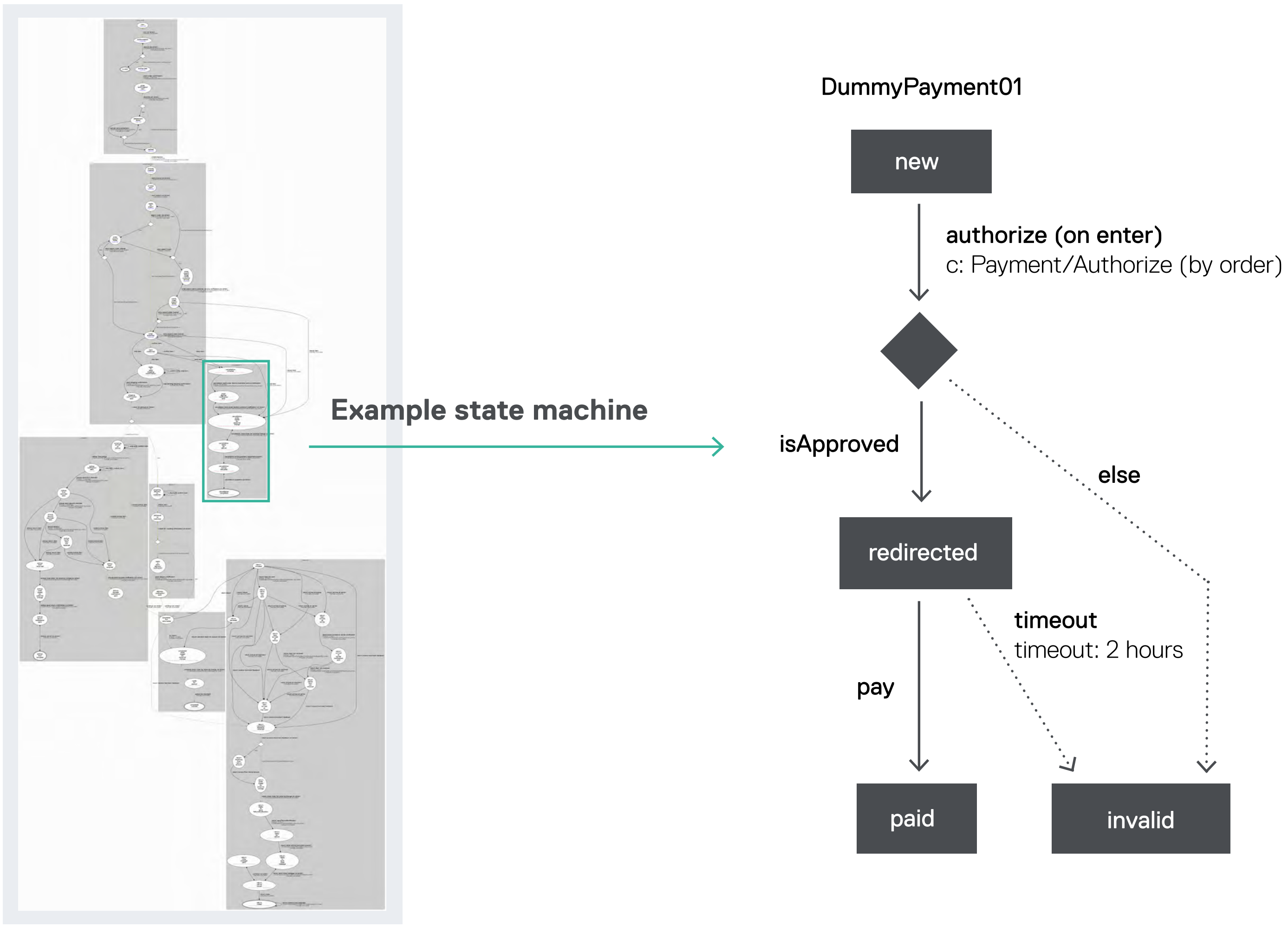


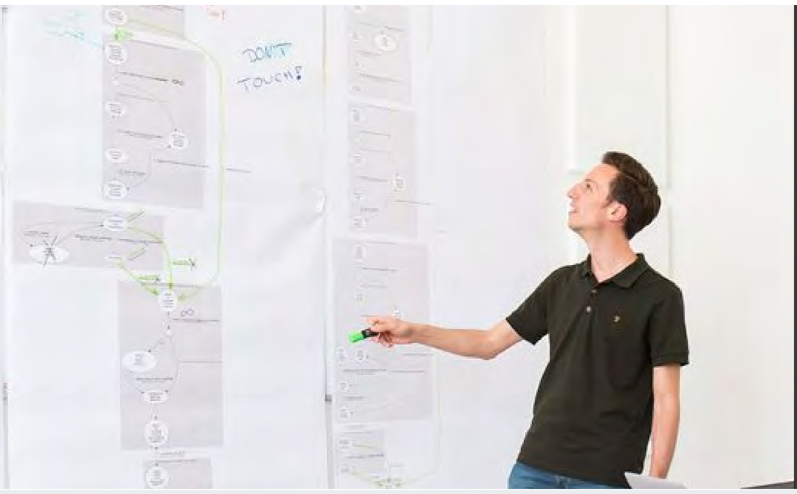
Full state machine

```
<?xml version="1.0"?>
<statemachine>

    <process name="DummyPayment01" main="true">

        <states>
            <state name="new"/>
            <state name="redirected"/>
            <state name="paid"/>
            <state name="invalid"/>
        </states>

        <transitions>

            <transition happy="true" condition="isApproved">
                <source>new</source>
                <target>redirected</target>
                <event>authorize</event>
            </transition>

            <transition>
                <source>new</source>
                <target>invalid</target>
                <event>authorize</event>
            </transition>

            <transition happy="true">
                <source>redirected</source>
                <target>paid</target>
                <event>pay</event>
            </transition>

            <transition>
                <source>redirected</source>
                <target>invalid</target>
                <event>timeout</event>
            </transition>

        </transitions>

        <events>
            <event name="authorize" onEnter=„true“
command="Payment/Authorize"/>
            <event name="pay"/>
            <event name="timeout" timeout="2 hours"/>
        </events>
    </process>

</statemachine>
```

Real world state machines are big

- Spryker state machine supports versions, so you can have different versions of the same state machine.
- One shop can have several state machines for different processes.
- State machine can have sub-processes allowing better management of big processes
- All the data of all the orders in a shop is stored in the shop's database, great source for robust BI analysis.





[State machine in action @Contorion](#)

„Die Website ist unsere #Baustelle" - Interview mit dem Contorion Projektmanagement.

Spryker

spryker.com hello@spryker.com @sprysys