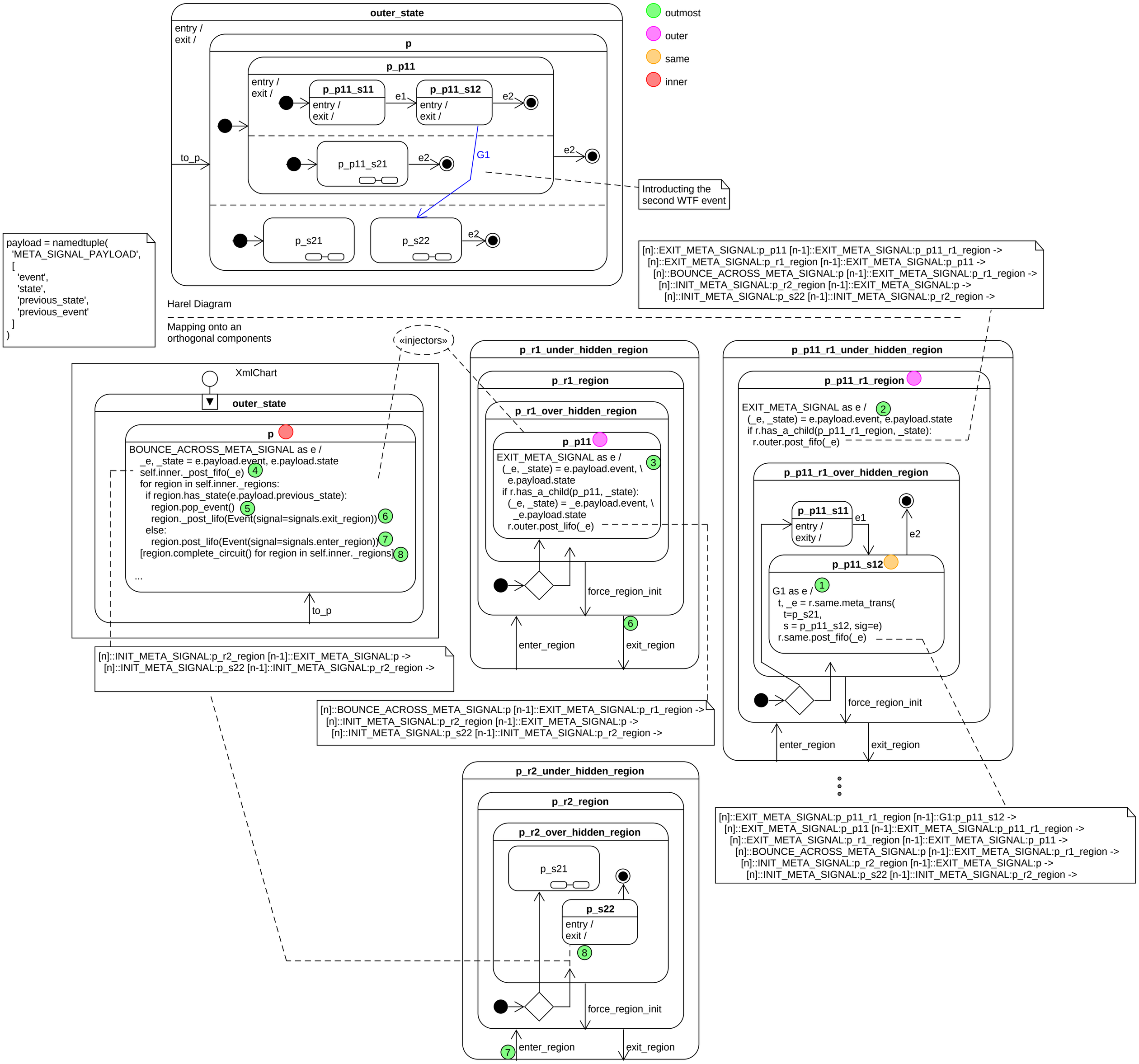
outer_state
entry /
exit /
p
p_p11
entry /
exit /
p_p11_s11
entry /
exit /
e1
p_p11_s12
entry /
exit /
e2
to_p
p_p11_s21
e2
G1
e2
p_s21
p_s22
e2
outmost
outer
same
inner
Introducting the
second WTF event
payload = namedtuple(
'META_SIGNAL_PAYLOAD',
[
'event',
'state',
'previous_state',
'previous_event'
]
)
[n]::EXIT_META_SIGNAL:p_p11 [n-1]::EXIT_META_SIGNAL:p_p11_r1_region ->
[n]::EXIT_META_SIGNAL:p_r1_region [n-1]::EXIT_META_SIGNAL:p_p11 ->
[n]::BOUNCE_ACROSS_META_SIGNAL:p [n-1]::EXIT_META_SIGNAL:p_r1_region ->
[n]::INIT_META_SIGNAL:p_r2_region [n-1]::EXIT_META_SIGNAL:p ->
[n]::INIT_META_SIGNAL:p_s22 [n-1]::INIT_META_SIGNAL:p_r2_region ->
Harel Diagram
Mapping onto an
orthogonal components
«injectors»
XmlChart
outer_state
p
BOUNCE_ACROSS_META_SIGNAL as e /
_e, _state = e.payload.event, e.payload.state
self.inner._post_fifo(_e) 4
for region in self.inner._regions:
if region.has_state(e.payload.previous_state):
region.pop_event() 5
region._post_lifo(Event(signal=signals.exit_region)) 6
else:
region.post_lifo(Event(signal=signals.enter_region)) 7
[region.complete_circuit() for region in self.inner._regions] 8
...
to_p
[n]::INIT_META_SIGNAL:p_r2_region [n-1]::EXIT_META_SIGNAL:p ->
[n]::INIT_META_SIGNAL:p_s22 [n-1]::INIT_META_SIGNAL:p_r2_region ->
p_r1_under_hidden_region
p_r1_region
p_r1_over_hidden_region
p_p11
EXIT_META_SIGNAL as e /
(_e, _state) = e.payload.event, \
e.payload.state
if r.has_a_child(p_p11, _state):
(_e, _state) = _e.payload.event, \
_e.payload.state
r.outer.post_lifo(_e) 3
force_region_init
6
enter_region
exit_region
[n]::BOUNCE_ACROSS_META_SIGNAL:p [n-1]::EXIT_META_SIGNAL:p_r1_region ->
[n]::INIT_META_SIGNAL:p_r2_region [n-1]::EXIT_META_SIGNAL:p ->
[n]::INIT_META_SIGNAL:p_s22 [n-1]::INIT_META_SIGNAL:p_r2_region ->
p_p11_r1_under_hidden_region
p_p11_r1_region
EXIT_META_SIGNAL as e / 2
(_e, _state) = e.payload.event, e.payload.state
if r.has_a_child(p_p11_r1_region, _state):
r.outer.post_fifo(_e)
p_p11_r1_over_hidden_region
p_p11_s11
entry /
exity /
e1
e2
p_p11_s12
G1 as e / 1
t, _e = r.same.meta_trans(
t=p_s21,
s = p_p11_s12, sig=e)
r.same.post_fifo(_e)
force_region_init
enter_region
exit_region
[n]::EXIT_META_SIGNAL:p_p11_r1_region [n-1]::G1:p_p11_s12 ->
[n]::EXIT_META_SIGNAL:p_p11 [n-1]::EXIT_META_SIGNAL:p_p11_r1_region ->
[n]::EXIT_META_SIGNAL:p_r1_region [n-1]::EXIT_META_SIGNAL:p_p11 ->
[n]::BOUNCE_ACROSS_META_SIGNAL:p [n-1]::EXIT_META_SIGNAL:p_r1_region ->
[n]::INIT_META_SIGNAL:p_r2_region [n-1]::EXIT_META_SIGNAL:p ->
[n]::INIT_META_SIGNAL:p_s22 [n-1]::INIT_META_SIGNAL:p_r2_region ->
p_r2_under_hidden_region
p_r2_region
p_r2_over_hidden_region
p_s21
p_s22
entry /
exit /
8
force_region_init
7
enter_region
exit_region