NewWave - Workflow engine

Sebastijan Kaplar, Miroslav Zarić, Gordana Milosavljević

University of Novi Sad

Faculty of Technical Sciences

NewWave

- Open-source workflow engine
- Natively written in Pharo
- In-development

Ideas

- Extensible workflow management system
- Facilitate easy specification and implementation of business logic
- Integration with other applications and tools
- Distributing NewWave over multiple Pharo images (enabling orchestration and communication between them)
- **Act independently**

Real life usage scenarios

- Interactive workflows
    - Steps in the workflow executed by one or more users
- Partially or completely automated workflows
    - Some tasks in the workflow can be delegated to the software component, service or other automated component
- IoT and BPM

Supported patterns

- Sequence (*sequential routing*)
- Conditional routing (*XOR-split*)
- Parallel routing (*AND-split*)
- Synchronization (*AND-join*)

Designing the workflow

- No graphical workflow editor/designer (for now)
- Specification of the workflow accomplished directly in the Pharo environment
- Inspection possible with Roassal

Sequence

Playground

Page

```
se := StartEvent new.
se description: 'StartEvent'.

t1 := BaseTask new.
t1 description: 'Task 1'.
t1 value: 'I am a simple task'.

ee := EndEvent new.
ee description: 'EndEvent'.

se addOutgoingEdge: t1.
t1 addOutgoingEdge: ee.

engine := WaveEngine initialNode: se.
engine startEngine.
```

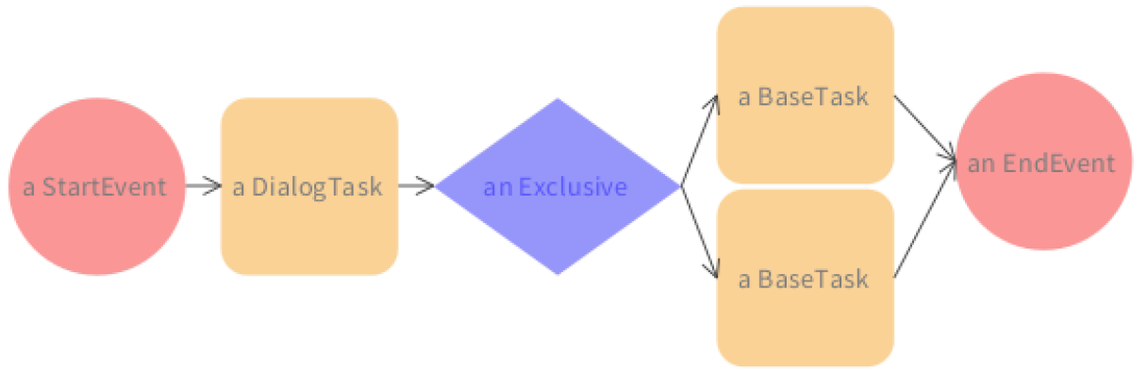
a RTView

View | Raw | Elements | Meta

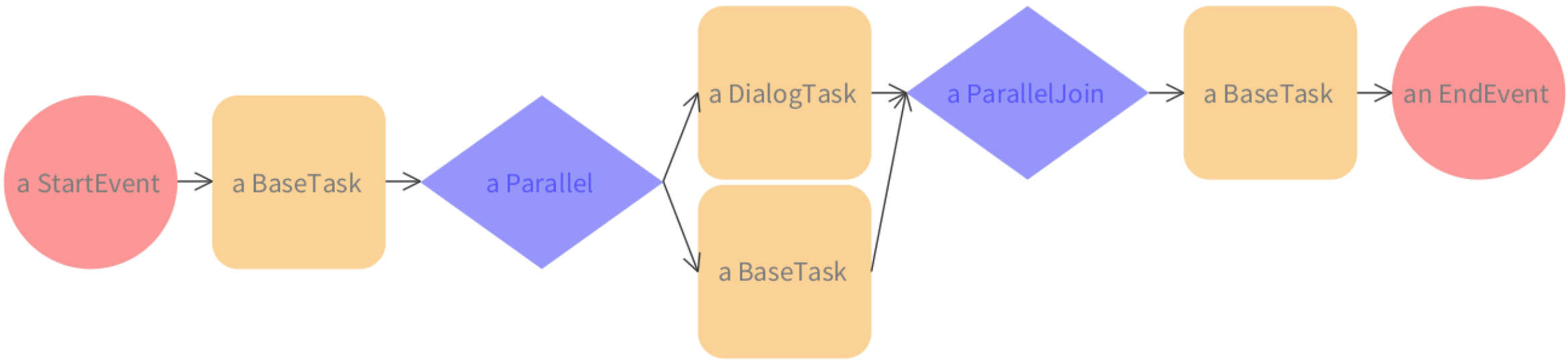
Sequential routing

```
se := StartEvent new.
se description: 'StartEvent'.

t1 := DialogTask new.
t1 description: 'Do you want to send a notification?'.

t2 := BaseTask new.
t2 description: 'Task 2'.
t2 value: 'Sending notification...'.

t3 := BaseTask new.
t3 description: 'Task 3'.
t3 value: 'Abandoning...'.

ee := EndEvent new.
ee description: 'EndEvent'.

split1 := Exclusive new.
split1 description: 'Split1'.

se addOutgoingEdge: t1.
t1 addOutgoingEdge: split1.
split1 addOutgoingEdge: t2 withCondition: [ :x | x = true ].
split1 addOutgoingEdge: t3 withCondition: [ :x | x = false ].
```

View | Raw | Elements | Meta



Parallel - Split/Join



Tasks

- BaseTask
- ScriptTask
- UserTask
- CustomTask

Events

- Supported events:
  - Start
  - End
  - Intermediate (Timer event, Announcement events)
  - Boundary events (controlling task execution)

Native Libraries

- TaskIt
- Announcements
- Roassal
- NeoJSON
- Teapot
- Mustache

Important Questions

- Refining the assignment mechanism
- DB Storage of process definitions and process instances
- UI for process execution environment
- Interacting with the engine

Interacting with the engine (How to)

- Creating a plugin that will send/receive data from the engine

- Data exchanged using REST via JSON/STON something else?

- Seaside/MDL or Teapot/Mustache?

Recent work/experimental

- Added Teapot server to the engine
- Serving tasks as JSON
- Plugin for obtaining tasks from server
- Rendering the form using mustache
- Submitting the results to server