DDS
Dynamic Deployment System

Anar Manafov, Andrey Lebedev
GSI Darmstadt
2014-03-21

DDS will be the deployment system for AlFa

DDS will be the PoD successor

An independent set of utilities and interfaces, which provide a dynamic distribution of different user processes by any given topology on any RMS.

Design Goals

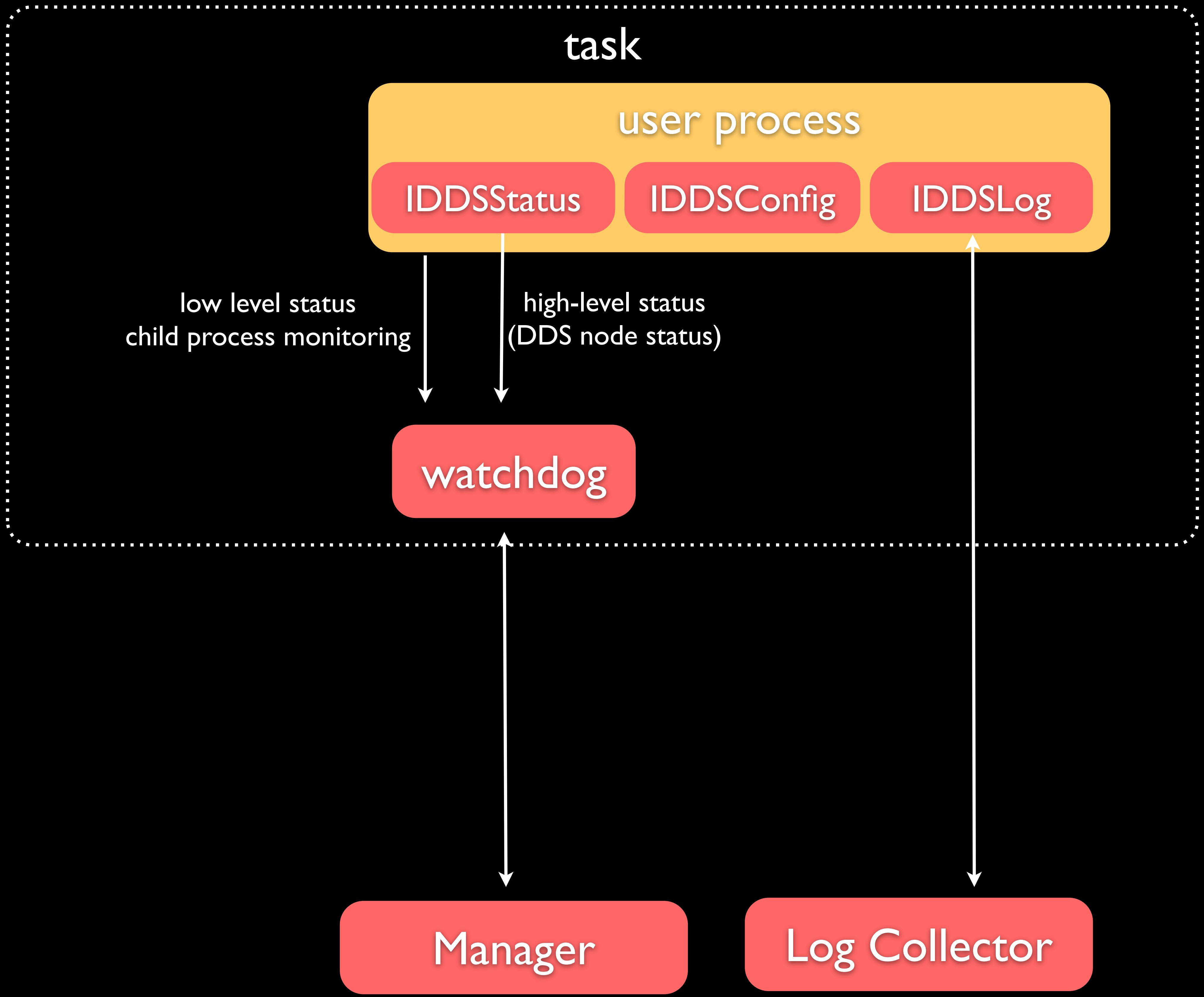
- deploy any task or a set of tasks,
- utilize any RMS,
- support workers behind FireWalls,
- secure execution of tasks (watchdog),
- support different topologies and task dependencies,
- provide an isolated execution,
- provide a central log engine.

Task

A task

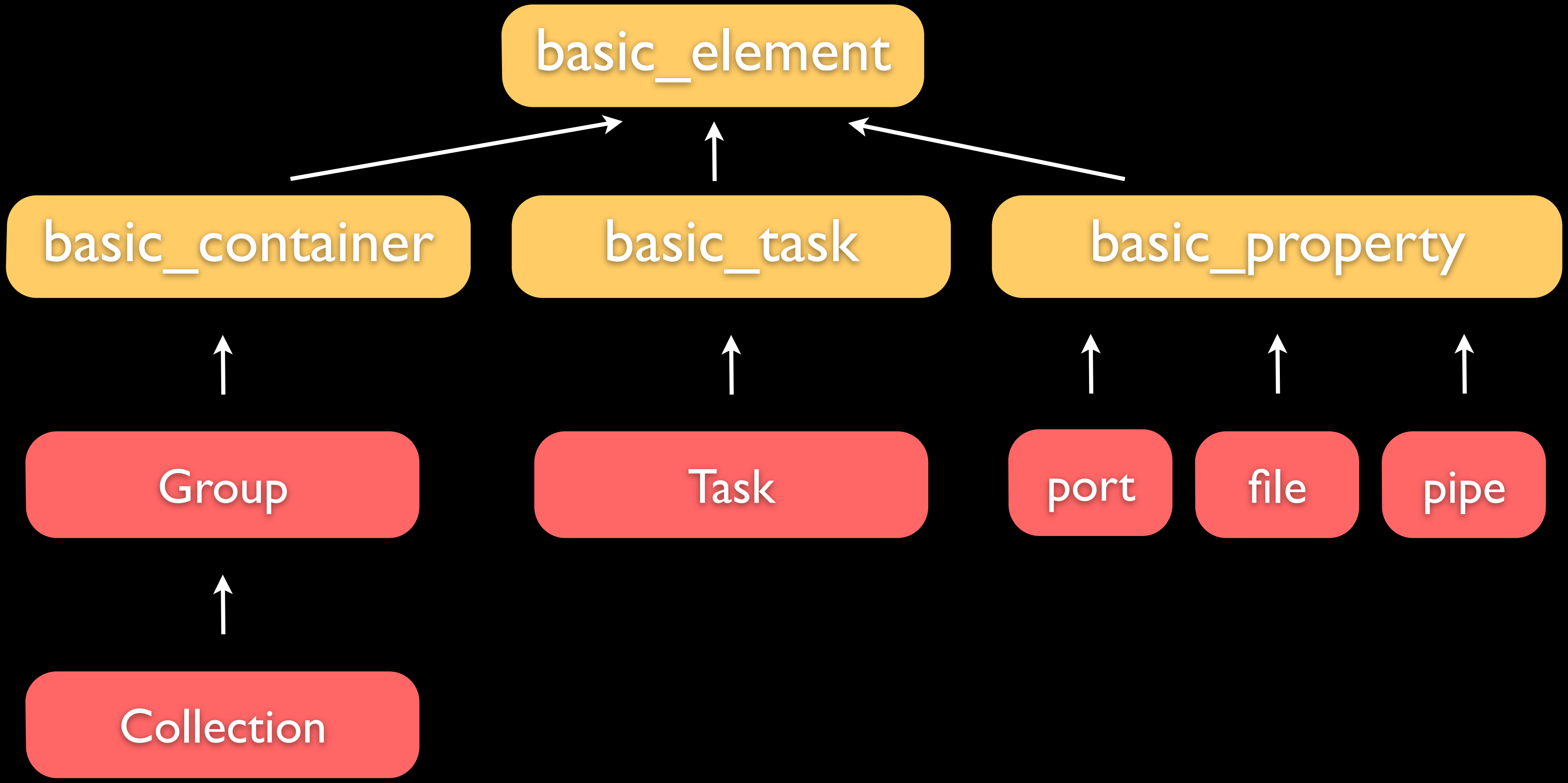
- is a single entity of the system,
- can be an executable or a script,
- is defined by a user with a set of props and rules,
- each task will have a dedicated DDS watchdog process.

DDS Task


task
user process
IDDSStatus
IDDSConfig
IDDSLog
low level status
child process monitoring
high-level status
(DDS node status)
watchdog
Manager
Log Collector

Topology and Topology language

Elements of the topology



```
<topology name="myTopology">

[… Definition of tasks, properties, and
collections …]

  <main name=“main">

[… Definition of the topology itself,
where also groups can be defined …]

  </main>

</topology>
```

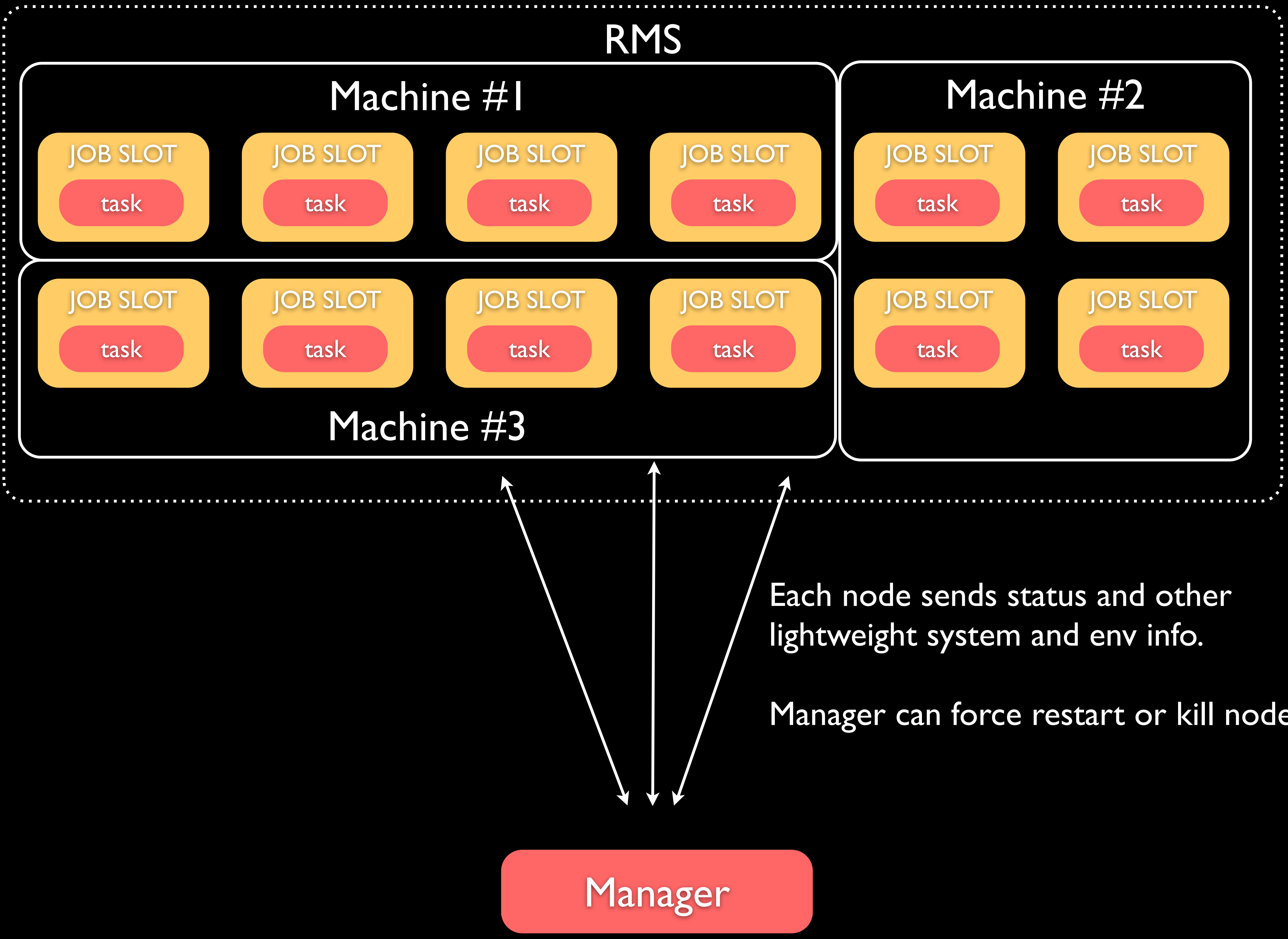
```
<topology name="my_PROOF_Topology">
  <port name="srv_port" min="20000" max="22000"/>
  <port name="wn_port" min="20000" max="22000"/>

  <task name="server" exec="proof.exe">
    <port name="wn_port"/>
    <port name="srv_port" server=yes/>
  </task>
  <task name="worker" exec="proof.exe" arg="-w">
    <port name="wn_port" server=yes/>
  </task>

  <main name="proof_cluster">
    <task name="server"/>
    <group name="group1" n="100" minRequired="1">
      <task name="worker"/>
    </group>
  </main>

</topology>
```

```
<topology name="myTopology">

[…]

   <collection name="collection1">
      <task name="task1"/>
      <task name="task2"/>
      <task name="task2"/>
   </collection>

   <collection name="collection2">
      <task name="task4"/>
      <task name="task5"/>
   </collection>

   <main name="main">
      <task name="task3"/>
      <collection name="collection1"/>
      <group name="group1" n="10" minRequired="1">
         <task name="task1"/>
         <collection name="collection1"/>
         <collection name="collection2"/>
      </group>
      <group name="group2" n="15" minRequired="3">
         <task name="task4"/>
         <collection name="collection1"/>
         <collection name="collection2"/>
      </group>
   </main>

</topology>
```

Examples Rules

- dependency rules
- start rules (for example if a time out is needed or before starting this node other node should be online)
- restart rules (what to do if a node died)
- I-am-busy rules (what to do if the node is too busy)
- ...

Topology



Each node sends status and other lightweight system and env info.

Manager can force restart or kill nodes

The Plan

Finish the first stable prototype in ~2 Month from now.

- Be able to parse and understand simple topologies with a limited number of properties (port, file).
- Be able to provide the same property from a set of tasks, array of properties.
- Provide an implementation of IDDSConfig.
- Release revised ssh plug-in, dds-ssh (former pod-ssh).