THE DYNAMIC DEPLOYMENT SYSTEM

DDS

ALICE Software and Computing Week,
2019-04-05

Anar Manafov, Andrey Lebedev,
GSI, Germany

DDS reached its design goals

DDS, in its current state, has reached all its major design goals and initial requirements.

We are working now to improv user experience and to tune different parts of the product in terms of performance and stability.

SOME GOODIES OF THE UPCOMING RELEASE

- Lightweight worker pkg.
  - Used by the localhost plug-in,
  - contains only essential files. Lib and exe are shared from the DDS dir,
  - deployment speed is x3 faster,
  - disk space usage reduced by a factor of 300 per agent (from 7-15MB to ~20KB per agent).
- DDS Custom Commands learned regex.
  - Regular expressions can be used to specifying destinations for CC.
- DDS agents learned to terminate child process of users' tasks.
  - On topology or task stop events, DDS agent will check and clean not only main processes of tasks, but also all their children.

There are many other features and bug fixes in the upcoming v2.4.
Check DDS release notes for more details.

FROM USER'S PERSPECTIVE

Topology

```
<topology id="myTopology">

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

  <main name="main">

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

  </main>

</topology>
```

Tools

dds-session
dds-into
dds-submit
dds-topology
dds-agent-cmd
dds-custom-cmd
dds-prep-worker
dds-stat
dds-test
dds-user-defaults

Intercom API

```
CIntercomService service;
CKeyValue keyValue(service);
CCustomCmd cCmd(service);

// Subscribe on error events
service.subscribeOnError([](EErrorCode _errorCode,
                            const string& _msg)
{
        …
});

// Subscribe on key update events
keyValue.subscribe([](const string& _propertyID,
                      const string& _key,
                      const string& _value)
{
        …
});

// Subscribe on delete key notifications
keyValue.subscribeOnDelete([](const string&
_propertyID,
                              const string& _key)
{
        …
 });

// Start listening to events we have subscribed on
service.start();
```

FROM USER'S PERSPECTIVE

Topology

+

Topology API

Tools

+

Tools API

Intercom API

TOPOLOGY API

- Core and public API,
- runtime and custom topologies,
- ability to modify and serialise topologies,
- methods for Tasks and Collections,
- methods for Properties,
- iterators for Tasks by a condition (for example, by a property name or type).

DDS ver. 2.3.21.gda36089

Main Page | Related Pages | Namespaces | Classes | Files

dds > topology_api > CTopology

dds::topology_api::CTopology Class Reference

`#include <Topology.h>`

Public Member Functions

	CTopology () Default constructor. More...
	~CTopology () Destructor. More...
void	**init** () Initializes default topology for DDS agent. More...
void	**init** (const std::string &_fileName) Initializes topology with the specified file without validation. More...
void	**init** (const std::string &_fileName, const std::string &_schemaFileName) Initializes topology with the specified file and validates against provided schema file. More...
CTopoGroup::Ptr_t	**getMainGroup** () const Returns shared pointer to the main group of the topology. More...
const STopoRuntimeTask &	**getRuntimeTaskById** (Id_t _id) const Returns runtime task by ID. More...
const STopoRuntimeCollection &	**getRuntimeCollectionById** (Id_t _id) const Returns runtime collection by ID. More...
const STopoRuntimeTask &	**getRuntimeTaskByIdPath** (const std::string &_idPath) const Returns runtime task by path. More...
const STopoRuntimeCollection &	**getRuntimeCollectionByIdPath** (const std::string &_idPath) const Returns runtime collection by path. More...
STopoRuntimeTask::FilterIteratorPair_t	**getRuntimeTaskIterator** (STopoRuntimeTask::Condition_t _condition=nullptr) const Returns runtime task filter iterator. More...
STopoRuntimeCollection::FilterIteratorPair_t	**getRuntimeCollectionIterator** (STopoRuntimeCollection::Condition_t _condition=nullptr) const Returns runtime collection filter iterator. More...

EXAMPLE: Get a number of required agents

```
CTopology topo();
topo.init("my_topo.xml");
const size_t nAgents = topo.getNofRequiredAgents();
```

EXAMPLE: Get a number of required agents

```
> ddd-topology --required-agents my_topo.xml
129
```

EXAMPLE: Get list of properties of the given task

```
CTopology topo;
topo.init();
uint64_t taskId = env_prop<EEnvProp::task_id>();
const STopoRuntimeTask& runtimeTask = topo.getRuntimeTaskById(taskId);
const CTopoProperty::PtrMap_t& properties = runtimeTask.m_task->getProperties();
```

TOOLS API

- Based on the DDS Custom Commands protocol.
- Basic commands, which can be combined to form complex requests.
- Implements commands to interact with:
  - Sessions (start/stop/info),
  - RMS Plug-ins (submit/info/list),
  - Status (server/agents info),
  - etc.
- Users can create custom CLI tools and use DDS from within C++ (compatible) projects.
- DDS CLI will be re-implemented using Tools API, instead of Core DDS Protocol.
  - It will significantly relax the Core Protocol,
  - It will simplify implementation of DDS CLI.
  - It will help to extend DDS CLI with minimum development resources.

FUTURE PLANS

- Main focus on the next stable release (v2.4):
  - New APIs tuning/polishing/tests,
  - stability improvements.
- v2.6 milestones.
- Integration with CBM and other FAIR experiments.

- Dev. Release DDS v2.3 (http://dds.gsi.de/download.html),
- DDS Home site: http://dds.gsi.de
- User's Manual & API doc.: http://dds.gsi.de/documentation.html
- Continuous integration: http://demac012.gsi.de:22001/waterfall
- Source Code: https://github.com/FairRootGroup/DDS https://github.com/FairRootGroup/DDS-user-manual https://github.com/FairRootGroup/DDS-web-site