DDS
Dynamic Deployment System

Anar Manafov, Andrey Lebedev
GSI Darmstadt
2014-06-27

Basic concepts

DDS:

- implements a single-responsibility-principle command line tool-set and APIs,
- treats users' tasks as black boxes,
- doesn't depend on RMS,
- doesn't require WNs to be pre-installed,
- deploys private facilities on demand with isolated sandboxes,
- provides a key-value properties propagation service for tasks,
- provides a rules based execution of tasks.

System's Capabilities

The system is capable to

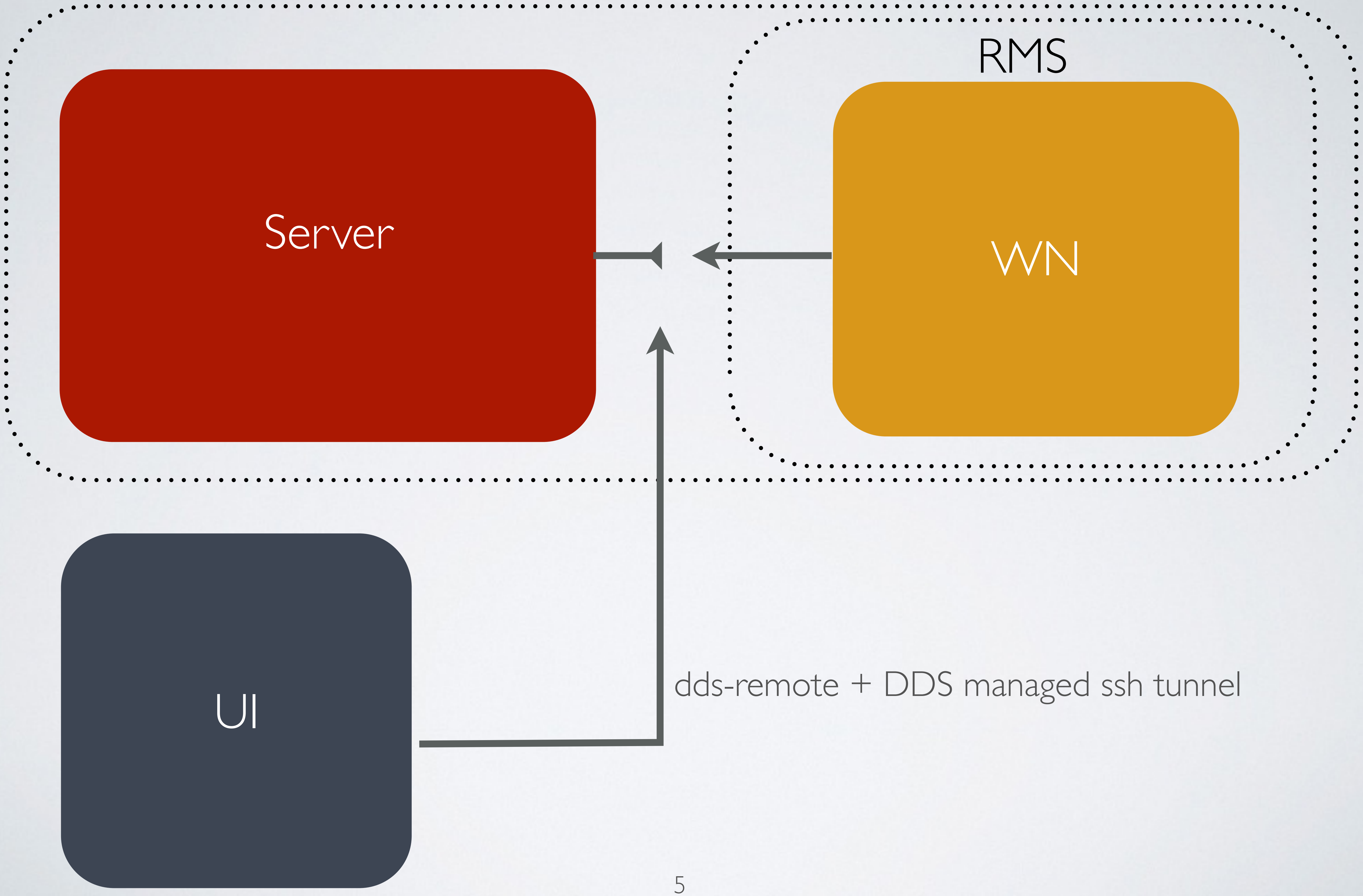
- deploy any executable/script (task) or a set of tasks,
- utilise any RMS,
- provide deployment via SSH (when no RMS is present),
- support workers behind FireWalls,
- secure execution of tasks (watchdog service),
- support different topologies and task dependencies (property propagation),
- provide an isolated execution,
- provide a central log engine.

The Contract

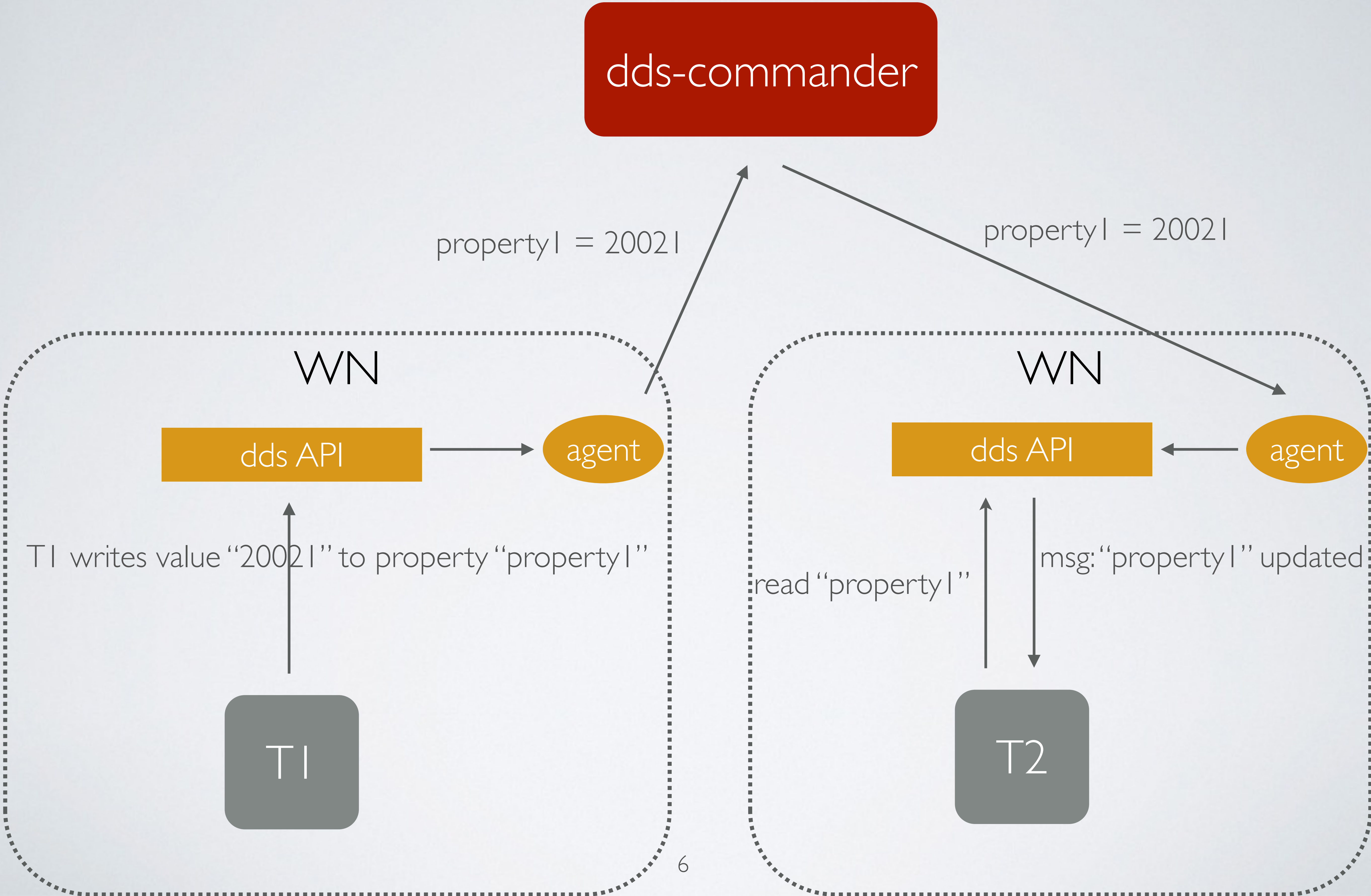
System takes so called a topology file as the input
(see. back-up slides for examples).

- currently XML. Can be any format supported by the boost property tree lib, even custom formats are possible.
- Users will be provided with a WEB GUI to create topos. Can be create manually as well.
- A given topo file is always parsed and verified before processing - this is the main reason to have verifiable file formats as the input.

Components of the system



key-value properties propagation



key-value properties propagation

Can't/don't want use DDS API for properties?!
No problem. You can access your config directly.

DDS will create a config file for each task - "<task name>.cfg", which can be monitored for changes and accessed directly.

Components of the system

RMS

Server
- dds-commander
- dds-user-defaults
- dds-user-defaults-lib
- dds-topology-lib
- dds-rms (RMS plug-ins)
- dds-ssh

WN
- dds-scaut
- dds-agent
- dds-user-defaults-lib
- dds-user-defaults

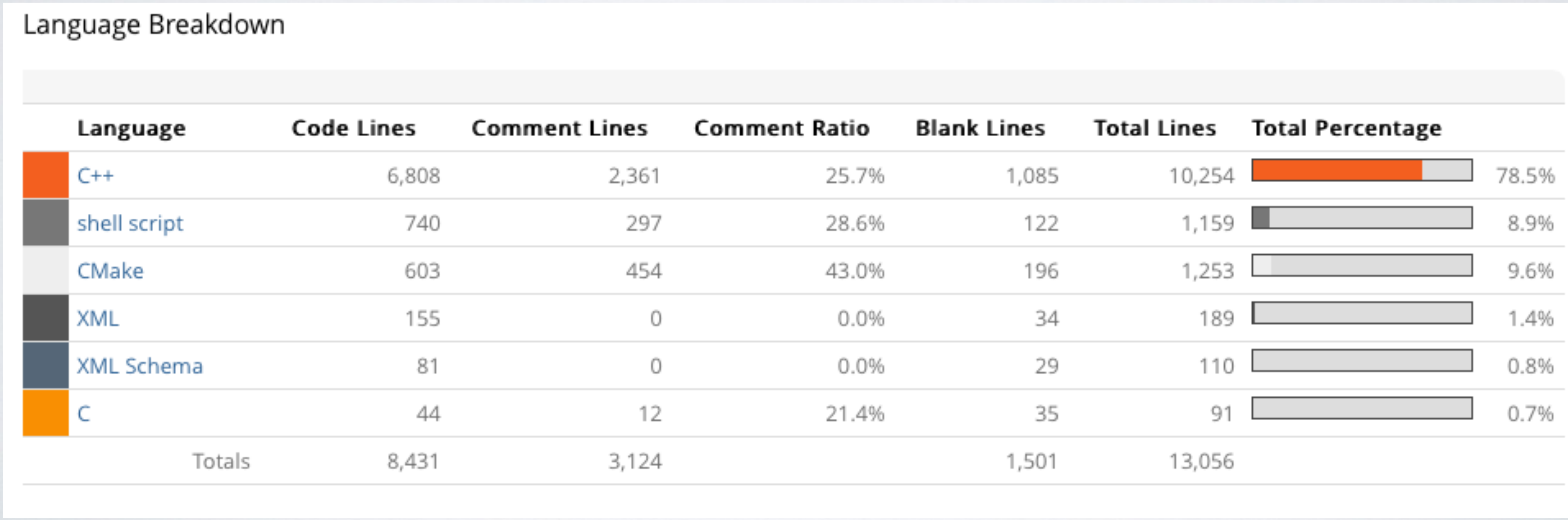
UI
- dds-remote
- dds-user-defaults
- ddf-info

Development status

Investigation, brainstorming, testing existing tools: **Sept, 2013**

Development phase start: **Feb, 2014**

2 active developers



Language Breakdown

	Language	Code Lines	Comment Lines	Comment Ratio	Blank Lines	Total Lines	Total Percentage
🟧	C++	6,808	2,361	25.7%	1,085	10,254	78.5%
⬛	shell script	740	297	28.6%	122	1,159	8.9%
⬜	CMake	603	454	43.0%	196	1,253	9.6%
⬛	XML	155	0	0.0%	34	189	1.4%
⬛	XML Schema	81	0	0.0%	29	110	0.8%
🟧	C	44	12	21.4%	35	91	0.7%
	Totals	8,431	3,124		1,501	13,056	

Current dependencies

- C++11 compatible compiler,
- cmake 2.6.2+
- boost 1.41+

Dev. state of the components

- dds-commander - 5%
- dds-agent - 5%
- ddf-info - 50%
- dds-scaut - 80%
- dds-ssh - 90%
- dds-remote - 90%

- dds-topology-lib - 100%
- ddb-user-defaults - 100%
- dds-user-defaults-lib - 100%
- dds-rms (RMS plug-ins) - 0%

Back-up slides

```
<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>
```

```xml
<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>
```