NETCONF a YANG

Ladislav Lhotka • lhotka@nic.cz • 29. listopadu 2014

Osnova

1. Proč NETCONF?
2. K čemu je datový model?
3. Základy jazyka YANG;
4. Turingův stroj - datový model;
5. Jak funguje NETCONF?
6. Knihovna *libnetconf*
7. NETCONF server pro Turingův stroj
8. Závěr

Proč NETCONF?

Máme přece své oblíbené CLI, konfigurační soubor, webové rozhraní, …

jenže

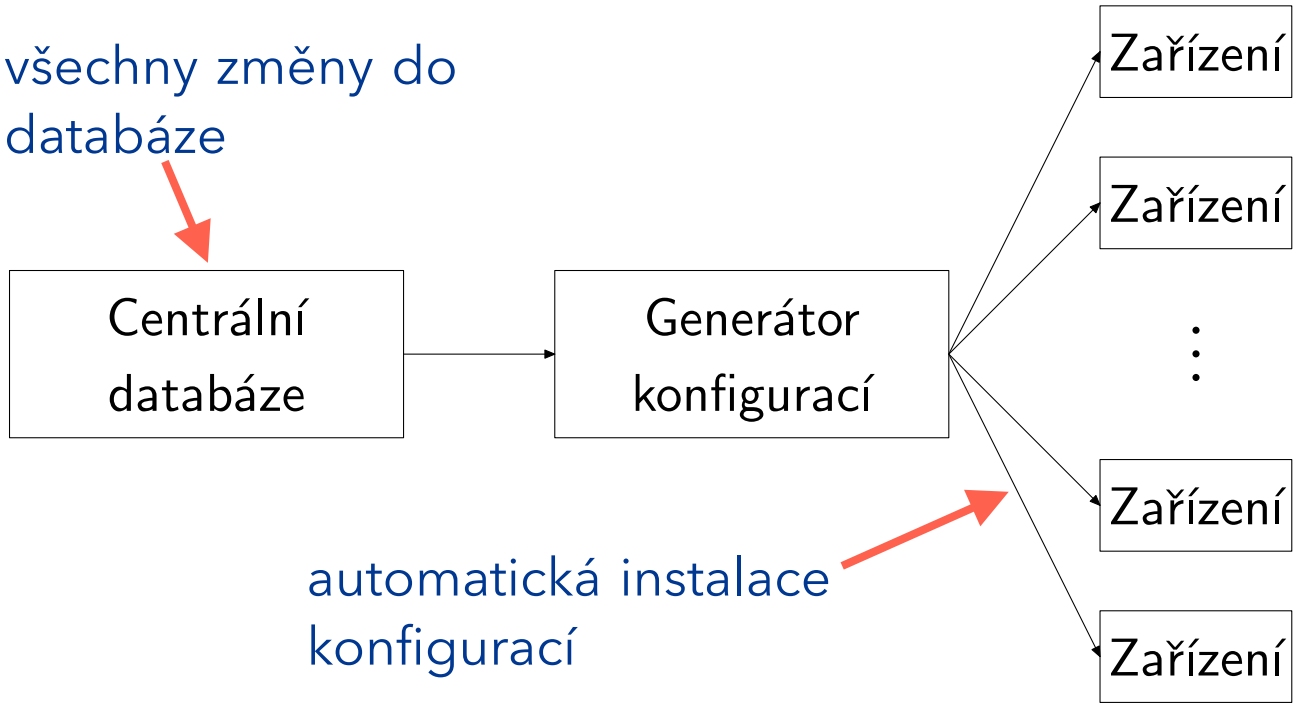
žádné z těchto rozhraní (obvykle) nepodporuje

- efektivní automatizaci,
- souběžný přístup více správců,
- konfigurační změny s volitelným rozlišením,
- obnovení předchozího stavu v případě problémů.

NETCONF toto vše umožňuje.

Náhradní řešení: Expect skripty, screenscraping.

Typický workflow



Protokol NETCONF

NETCONF je nenáročný klient-server protokol definovaný v RFC 6241. Základní vlastnosti:

- rozlišují se konfigurační a stavová data;
- interakce typu dotaz-odpověď anebo (volitelné) asynchronní notifikace;
- metody pro čtení a editaci dat;
- protokolové operace i vlastní data se kódují v XML.

Alternativou k NETCONFu je RESTCONF, který využívá standardní metody HTTP a kromě XML podporuje i JSON.

K čemu je datový model?

Datový model definuje *schéma* konfiguračních a stavových dat, ale také jejich sémantiku a omezení (business rules).

Datový model může být použit pro formální validaci, ale také jako dokumentace pro vývojáře, který datový model implementuje.

Pro XML existuje několik jazyků pro popis schématu dokumentu:

- Document Type Definition (DTD),
- W3C XML Schema,
- Document Schema Definition Languages (DSDL),
  - RELAX NG: gramatika, datové typy,
  - Schematron: sémantická pravidla,
  - Document Schema Renaming Language (DSRL): doplnění defaultů a jiné úpravy.

Jazyk YANG

Požadavky:

- společná specifikace syntaxe (schématu), datových typů i sémantických pravidel;
- *čitelnost*, preference autorů datových modelů a softwarových vývojářů jsou až sekundární;
- modularita a škálovatelnost;
- zachování vztahu k zavedeným XML technologiím (jazykům pro popis schémat).

YANG 1.0 byl definován v RFC 6020, pracuje se na verzi 1.1.

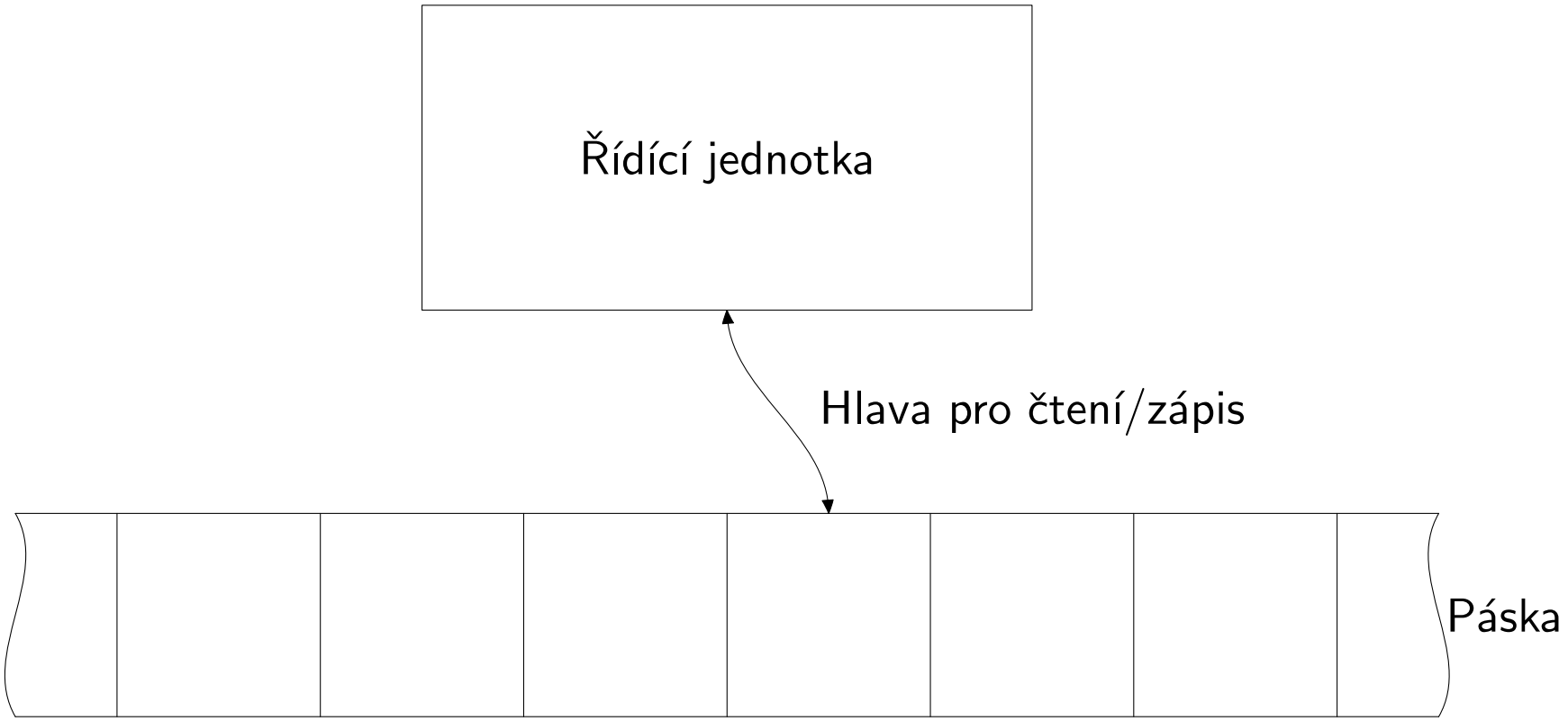
Základy jazyka YANG

Datový model se skládá z jednoho či více YANG *modulů*, každý modul může být dále rozdělen do *submodulů*.

Struktura dat je hierarchická, datový strom může obsahovat pět typů uzlů:

- *leaf* – skalární parametr s definovaným typem;
- *container* – vnitřní uzel stromu, obsahuje další uzly;
- *leaf-list* – seznam skalárních parametrů stejného typu;
- *list* – seznam kontejnerů stejné struktury;
- *anyxml* – libovolný fragment XML, bez schématu.

Turingův stroj



YANG modul: záhlaví, metadata

```
module turing-machine {
  namespace "http://example.net/turing-machine";
  prefix "tm";
  description
    "Data model for the Turing Machine.";
  revision 2013-12-27 {
    description
      "Initial revision.";
  }
```

URI a prefix pro XML

YANG modul: datové typy

```
typedef tape-symbol {
  type string {
    length "0..1";
  }
  description
    "Type of symbols appearing in tape cells.
     A blank is represented as an empty string where necessary.";
}

typedef head-dir {
  type enumeration {
    enum left;
    enum right;
  }
  default "right";
  description
    "Possible directions for moving the read/write head, one cell
     to the left or right (default).";
}
```

omezení délky

implicitní hodnota

YANG modul: groupings

```
grouping tape-cells {
  description
    "The tape of the Turing Machine is represented as a sparse
     array.";
  list cell {
    key "coord";
    description
      "List of non-blank cells.";
    leaf coord {
      type cell-index;
      description
        "Coordinate (index) of the tape cell.";
    }
    leaf symbol {
      type tape-symbol {
        length "1";
      }
      description
        "Symbol appearing in the tape cell.
         Blank (empty string) is not allowed here because the
         'cell' list only contains non-blank cells.";
    }
  }
}
```

dodatečná restrikce typu

YANG modul: stavová data

```
container turing-machine {
  description
    "State data and configuration of a Turing Machine.";
  leaf state {
    type state-index;
    config "false";
    mandatory "true";
    description
      "Current state of the control unit.
       The initial state is 0.";
  }
  leaf head-position {
    type cell-index;
    config "false";
    mandatory "true";
    description
      "Position of tape read/write head.";
  }
  container tape {
    config "false";
    description
      "The contents of the tape.";
    uses tape-cells;
  }
```

YANG modul: konfigurační data

```
container transition-function {
  description
    "The Turing Machine is configured by specifying the
     transition function.";
  list delta {
    key "label";
    unique "input/state input/symbol";
    description
      "The list of transition rules.";
    leaf label {
      type string;
      description
        "An arbitrary label of the transition rule.";
    }
    container input {
      …
    }
    container output {
      …
    }
  }
}
```

klíč seznamu

unikátní kombinace

YANG modul: vlastní RPC metody

```
rpc initialize {
  description
    "Initialize the Turing Machine as follows:
     1. Put the control unit into the initial state (0).
     2. Move the read/write head to the tape cell with coordinate
        zero.
     3. Write the string from the 'tape-content' input parameter to
        the tape, character by character, starting at cell 0. The
        tape is othewise empty.";
  input {
    leaf tape-content {
      type string;
      default "";
      description
        "The string with which the tape shall be initialized. The
         leftmost symbol will be at tape coordinate 0.";
    }
  }
}
rpc run {
  description
    "Start the Turing Machine operation.";
}
```

YANG modul: notifikace

```
notification halted {
  description
    "The Turing Machine has halted. This means that there is no
     transition rule for the current state and tape symbol.";
  leaf state {
    type state-index;
    mandatory "true";
    description
      "The state of the control unit in which the machine has
       halted.";
  }
}
```

Práce s modulem

Free software: *pyang*.

- kontrola modulu

```
$ pyang turing-machine.yang
```

- UML diagram

```
$ pyang -f uml -o tm.uml turing-machine.yang \
> --uml-no=stereotypes,annotation,typedef

$ plantuml tm.uml
```

- datový strom (ASCII art)

```
$ pyang -f tree turing-machine.yang
module: turing-machine
   +--rw turing-machine
      +--ro state                    state-index
      +--ro head-position            cell-index
      +--ro tape
      |  +--ro cell* [coord]
      |     +--ro coord     cell-index
      |     +--ro symbol?   tape-symbol
      +--rw transition-function
         +--rw delta* [label]
            +--rw label     string
            +--rw input
            |  +--rw state     state-index
            |  +--rw symbol    tape-symbol
            +--rw output
               +--rw state?       state-index
               +--rw symbol?      tape-symbol
               +--rw head-move?   head-dir
rpcs:
   +---x initialize
   |  +---w input
   |     +---w tape-content?   string
   +---x run
notifications:
   +---n halted
      +--ro state    state-index
```

- kostra konfigurace

```
$ pyang -f sample-xml-skeleton turing-machine.yang \
> --sample-xml-skeleton-doctype=config \
> --sample-xml-skeleton-annotations | \
>  xmllint -o turing-machine-config.xml --format -
```

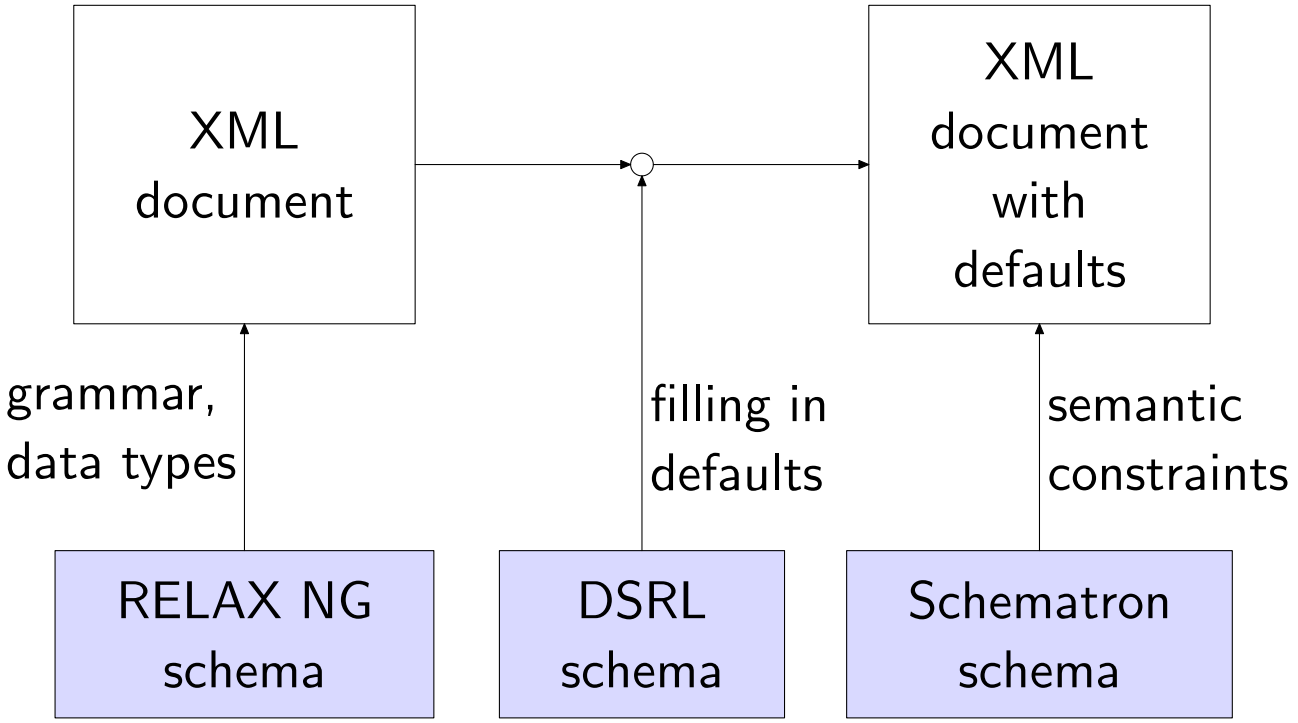
```
<config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <turing-machine xmlns="http://example.net/turing-machine">
    <transition-function>
      <delta>
        <!-- # entries: 0.. -->
        <label>
          <!-- type: string -->
        </label>
        <input>
          <state>
            <!-- type: state-index -->
          </state>
          <symbol>
            <!-- type: tape-symbol -->
          </symbol>
        </input>
        <output>
          <state>
            <!-- type: state-index -->
          </state>
          <symbol>
            <!-- type: tape-symbol -->
          </symbol>
        </output>
      </delta>
    </transition-function>
  </turing-machine>
</config>
```

- generování DSDL schémat (RELAX NG, Schematron, DSRL)

```
$ yang2dsdl -t config turing-machine.yang
$ trang -I rng -O rnc turing-machine-config.rng turing-machine-config.rnc
```

kompaktní syntaxe RELAX NG

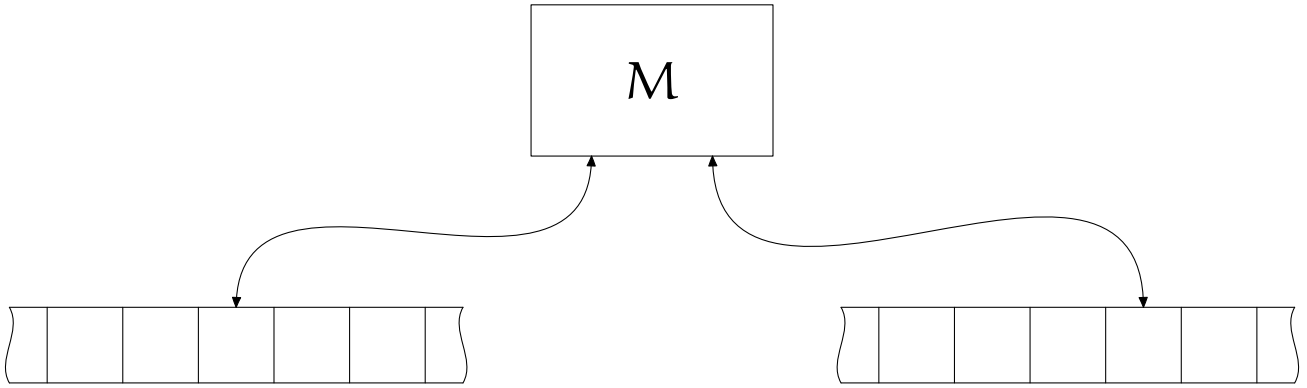
Validace pomocí DSDL schémat




```
$ yang2dsdl -s -j -t config -b turing-machine \
> -v turing-machine-config.xml
```

Nezávislé rozšíření datového modelu

Turingův stroj se dvěma páskami



Rozšiřující modul

```
module second-tape {
  namespace "http://example.net/turing-machine/tape-2";
  prefix "t2";
  import turing-machine {
    prefix "tm";
  }
  …
}
```

vlastní URI

přístup k pův. modulu

Rozšíření stavových dat


```
augment "/tm:turing-machine" {
  description
    "State data for the second tape.";
  leaf head-position-2 {
    config "false";
    type tm:cell-index;
    description
      "Head position of the second tape.";
  }
  container tape-2 {
    description
      "Contents of the second tape.";
    config "false";
    uses tm:tape-cells;
  }
}
```

Rozšíření konfigurace

```
augment
  "/tm:turing-machine/tm:transition-function/tm:delta/tm:input" {
  description
    "A new input parameter.";
  leaf symbol-2 {
    type tm:tape-symbol;
    description
      "Symbol read from the second tape.";
  }
}
```

```
augment
  "/tm:turing-machine/tm:transition-function/tm:delta/tm:output" {
  description
    "New output parameters.";
  leaf symbol-2 {
    type tm:tape-symbol;
    description
      "Symbol to be written to the second tape. If this leaf is not
       present, the symbol doesn't change.";
  }
  leaf head-move-2 {
    type tm:head-dir;
    description
      "Move the head on the second tape one cell to the left or
       right.";
  }
}
```

Rozšíření RPC metody

```
augment "/tm:initialize/tm:input" {
  description
    "A new RPC input parameter.";
  leaf tape-content-2 {
    type string;
    description
      "Initial content of the second tape.";
  }
}
```

Standardy

NETCONF

- RFC 6241: *Network Configuration Protocol (NETCONF).*
- RFC 5277: *NETCONF Event Notifications.*
- RFC 6244: *An Architecture for Network Management Using NETCONF and YANG.*
- draft-ietf-netconf-restconf-03: *RESTCONF Protocol.*

YANG

- RFC 6020: *YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF).*
- RFC 6087: *Guidelines for Authors and Reviewers of YANG Data Model Documents.*
- RFC 6110: *Mapping YANG to Document Schema Definition Languages and Validating NETCONF Content.*
- draft-ietf-netmod-yang-json-01: *JSON Encoding of Data Modeled with YANG.*

Datové modely

- RFC 7223: *A YANG Data Model for Interface Management.*
- RFC 7277: *A YANG Data Model for IP Management.*
- RFC 7317: *A YANG Data Model for System Management.*
- draft-ietf-netmod-routing-cfg-16: *A YANG Data Model for Routing Management.*

Další odkazy

- NETCONF Working Group

  `http://datatracker.ietf.org/wg/netconf/documents/`

- NETMOD Working Group

  `http://datatracker.ietf.org/wg/netmod/documents/`

- pyang

  `https://code.google.com/p/pyang/`

- NETCONF Central

  `http://www.netconfcentral.org/`

- YANG Central

  `http://www.yang-central.org/`

- Nástroje pro práci s YANGem, datové modely

  `https://gitlab.labs.nic.cz/labs/yang-tools`

- Document Schema Definition Languages (DSDL)

  `http://www.dsdl.org/`