gempa

SCUG 2013 Potsdam

SeisComP3 Seattle configuration and administration

Jan Becker, Marcelo Bianchi, Andres Heinloo

gempa GmbH & GFZ Potsdam

January 15, 2013

Outline

gempa

Design goals

Design goals

gempa

- Unified configuration framework that can be used by current and future applications
- Support default configuration to show a user how a configuration file looks like and to use it as a template
- Support user configuration that will work regardless of any automated or GUI driven tool that might destroy content it doesn't understand
- Self-describing module configurations to enable the development of configuration tools
- Remove shell/bash scripts to make all this more portable and less error prone (e.g. bash eval)
- Enable automatic generation of module and binding documentation
- Extensibility: new applications or plugins
- Facilitate the synchronization of two systems configuration-wise (or only parts of it like processing)
- Better view into system configuration

Directory structure

Directory structure

gempa

```
seiscomp3
├── bin/ ..... User binaries
├── etc/ ..... Application configurations
│   ├── descriptions/ ..... XML module descriptions
│   ├── defaults/ ..... Application default configurations
│   ├── init/ ..... Init/config scripts
│   ├── inventory/ ..... Inventory XML files
│   └── key/ ..... Station key files and bindings
├── include/ ..... SDK includes
├── lib/ ..... Common library directory
├── sbin/ ..... System/service binaries
├── share/ ..... Data directory
└── var/
    ├── run/ ..... run and pid files of applications
    ├── log/ ..... start log of applications
    └── lib/ ..... volatile application data
```

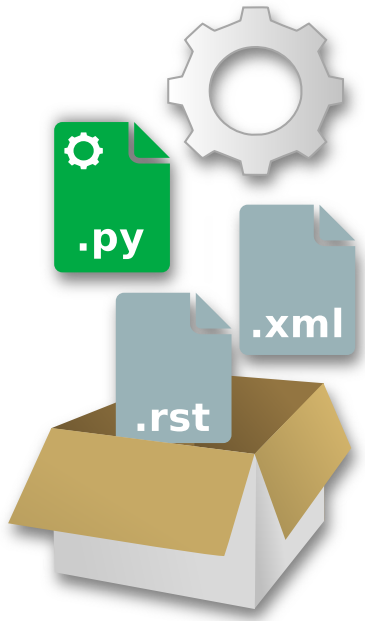
Module

Module

gempa

A module is defined by:

- a binary or script, **mandatory**, called by the modules init script
- an init script, **mandatory**, called by *seiscomp* to start/stop the module
- a description file, optional, used by *scconfig*
- its documentation, optional, used by the documentation generator



scautopick package example

```
bin/scautopick
etc/default/scautopick.cfg
etc/init/scautopick.py
etc/descriptions/scautopick.xml
```

Init scripts

Init scripts

gempa

- Python scripts installed under etc/init
- Called by *seiscomp*
- Can implement *seiscomp* events such as start, stop, check, update-config, setup, print, ...
- Required for each application, e.g. *seiscomp restart scautopick*
- Portable and easy to maintain
- Template available for new trunk modules

```
import seiscomp3.Kernel

class Module(seiscomp3.Kernel.Module):
  def __init__(self, env):
    seiscomp3.Kernel.Module.__init__(self, env, env.moduleName(__file__))

  def supportsAliases(self):
    # The default handler does not support aliases
    return True
```

Init scripts (cont.)

gempa

- Registers a module by name
- Two types: Module and CoreModule
- New enabled/disabled state which defines if a module should be started by *seiscomp start*, core modules are enabled by definition
- After a fresh installation all modules are disabled by default (spread and scmaster are core modules and as such enabled by definition)

Enable basic processing

```
sysop@host:~$ seiscomp3/bin/seiscomp enable scautopick scautoloc scamp scmag
scevent
enabled scautopick
enabled scautoloc
enabled scamp
enabled scmag
enabled scevent
```

Descriptions

Descriptions

gempa

Module descriptions are a fundamental concept to facilitate and to validate configuration. It furthermore pushes developers to document their modules.

- XML descriptions of modules, plugins or bindings
- Parsed from etc/descriptions/
- Contain general description, configuration parameters with type, default value and documentation
- Separation of control scripts and data model $\rightarrow$ enable generic configurator frontend such as *scconfig*
- Allows generation of manpages, HTML pages, default cfg files, ... from a single source

The benefit is not just a documentation that nobody reads: it makes *scconfig* to support your module configuration in a user friendly and convenient way!

Description Layout

gempa

```
<?xml version="1.0" encoding="UTF-8"?>
<seiscomp>
  <module name="..." category="..." standalone="...">
    <description>...</description>
    <configuration>...</configuration>
  </module>
  <binding name="..." module="..." category="...">
    <description>...</description>
    <configuration>...</configuration>
  </binding>
  <plug-in name="...">
    <extends>...</extends>
    <description>...</description>
    <configuration>...</configuration>
  </plug-in>
</seiscomp>
```

"module", "binding" and "plugin" elements can be split across multiple files to allow user contributions without touching shipped files.

Module example: scevent

gempa

```
<?xml version="1.0" encoding="UTF-8"?>
<seiscomp>
  <module name="scevent" category="Processing">
    <description>Associates Origins to Events or forms new Events if no suitable match is found.
                 Selects preferred magnitude.</description>
    <configuration>
      <parameter name="eventIDPrefix" type="string">
        <description>Prefix for all Event IDs</description>
      </parameter>
      <group name="eventAssociation">
        <parameter name="minimumMagnitudes" type="int" default="4">
          <description>Minimum number of station magnitudes referenced to a network magnitude
                       to become a preferred magnitude.</description>
        </parameter>
        ...
      </group>
    </configuration>
  </module>
</seiscomp>
```

Plugin example: NonLinLoc

gempa

```
<?xml version="1.0" encoding="UTF-8"?>
<seiscomp>
  <plug-in name="NonLinLoc">
    <extends>global</extends>
    <description>NonLinLoc locator wrapper plugin for SeisComP.
                 NonLinLoc was written by Anthony Lomax (http://alomax.free.fr/nlloc).</description>
    <configuration>
      <group name="NonLinLoc">
        <parameter name="publicID" type="string" default="NLL.@time/%Y%m%d%H%M%S.%f@.@id@">
          <description>PublicID creation pattern for an origin created by NonLinLoc.</description>
        </parameter>
        <parameter name="controlFile" type="path">
          <description>The default NonLinLoc control file to use.</description>
        </parameter>
        ...
      </group>
    </configuration>
  </plug-in>
</seiscomp>
```

Bindings example: Seedlink

gempa

- General binding definition to configure a station for Seedlink

```
<?xml version="1.0" encoding="UTF-8"?>
<seiscomp
  <binding module="seedlink">
    <configuration>
      <parameter name="access" type="list:string" default="0.0.0.0/0">
        <description>List of IP addresses or IP address/mask pairs,
        separated by spaces and/or commas.</description>
      </parameter>
      <parameter name="proc" type="string">
        <description>Name of the proc object (defined in streams.xml); used for
        processing raw streams (streams submitted by a plug-in as raw samples).</description>
      </parameter>
    </configuration>
  </binding>
</seiscomp>
```

Bindings example: Seedlink:chain

gempa

- Description of chain_plugin configurable for a station
- No fixed set of plugin parameter names (as in prior versions)
- Documentation of parameters in scope of plugin

```
<?xml version="1.0" encoding="UTF-8"?>
<seiscomp>
  <binding module="seedlink" name="chain" category="plugin">
    <description>Seedlink server (TCP/IP)</description>
    <configuration>
      <parameter name="address" type="host-with-port">
        <description>Address of the remote server in hostname:port format.</description>
      </parameter>
      <parameter name="selectors" type="list:string">
        <description>List of stream selectors. If left empty all available
        streams will be requested. See slinktool manpage for more information.</description>
      </parameter>
    </configuration>
  </binding>
</seiscomp>
```

Bindings example: Seedlink:q330

gempa

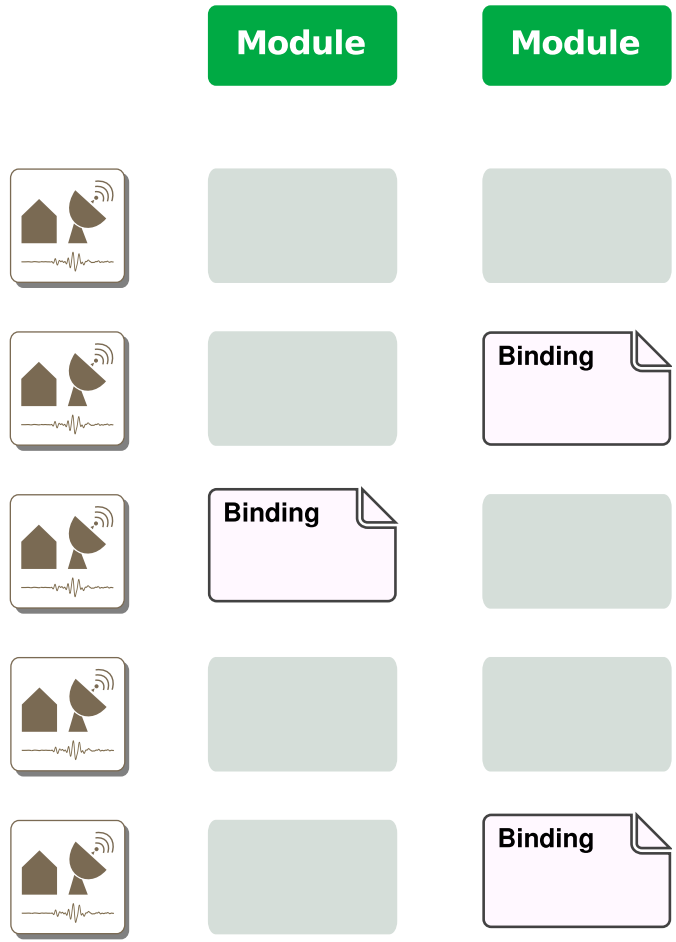
- Description of q330_plugin configurable for a station

```
<?xml version="1.0" encoding="UTF-8"?>
<seiscomp>
  <binding module="seedlink" name="q330" category="plugin">
    <description>Quanterra Q330 (UDP/IP)</description>
    <configuration>
      <parameter name="port" type="int" default="5330">
        <description>UDP port to receive data packets.</description>
      </parameter>
      <parameter name="slot" type="int" default="1"/>
      <parameter name="serial" type="string" default="0x0100000123456789"/>
      <parameter name="auth" type="string" default="0x00"/>
      ...
    </configuration>
  </binding>
</seiscomp>
```

Bindings

Bindings

- New term for the current package key files, e.g. acquisition/key
- **Binding** ≡ pkg/key/station_*_*
- **Binding profile** ≡ pkg/key/profile_*
- **One** binding configures **one** station for **one** module
- Binding configuration is using the same configuration format like modules



Bindings (cont.)

gempa

- Support for global and application specific database station configurations

```
# file: etc/key/global/profile_00HH
detecStream = HH
detecLocid = 00
```

```
# file: etc/key/scautopick/profile_tele
detecFilter = "RMHP(10)>>ITAPER(30)>>BW(4,0.7,2)>>STALTA(2,80)"
trigOn = 3
trigOff = 1.5
timeCorr = -0.8
```

- Separation of station configuration and meta-data: bindings are part of the module configuration

```
# file: etc/key/station_AB_CDEF
seedlink:geofon  # etc/key/seedlink/profile_geofon
global:00HH      # etc/key/global/profile_00HH
scautopick:tele  # etc/key/scautopick/station_AB_CDEF
```

gempa

Configuration of special plugin parameters map nicely to the database schema

```
detecStream = BH
trigOn = 3
trigOff = 1.5
# Set MLh.ClippingThreshold to 1E7 counts.
MLh.ClippingThreshold = 10000000
```

would end up in the database as

```
Parameter(name="detecStream", value="BH")
Parameter(name="trigOn", value="3")
Parameter(name="trigOff", value="1.5")
Parameter(name="MLh.ClippingThreshold", value="10000000")
```

Easy to understand and to document!

Bindings (cont.)

gempa

A plugin description file of the MLh amplitude plugin and its configuration options looks like this:

```
<?xml version="1.0" encoding="UTF-8"?>
<seiscomp>
        <binding name="MLh" module="global">
                <description>...</description>
                <configuration>
                        <group name="MLh">
                                <parameter name="maxavg" type="string" default="max">
                                        <description>...</description>
                                </parameter>
                                <parameter name="ClippingThreshold" type="double">
                                        <description>...</description>
                                </parameter>
                                <parameter name="params" type="string">
                                        <description>...</description>
                                </parameter>
                        </group>
                </configuration>
        </binding>
</seiscomp>
```

Metadata aka Inventory

Metadata aka Inventory

gempa

Inventory management has changed completely from release Zurich and prior versions!

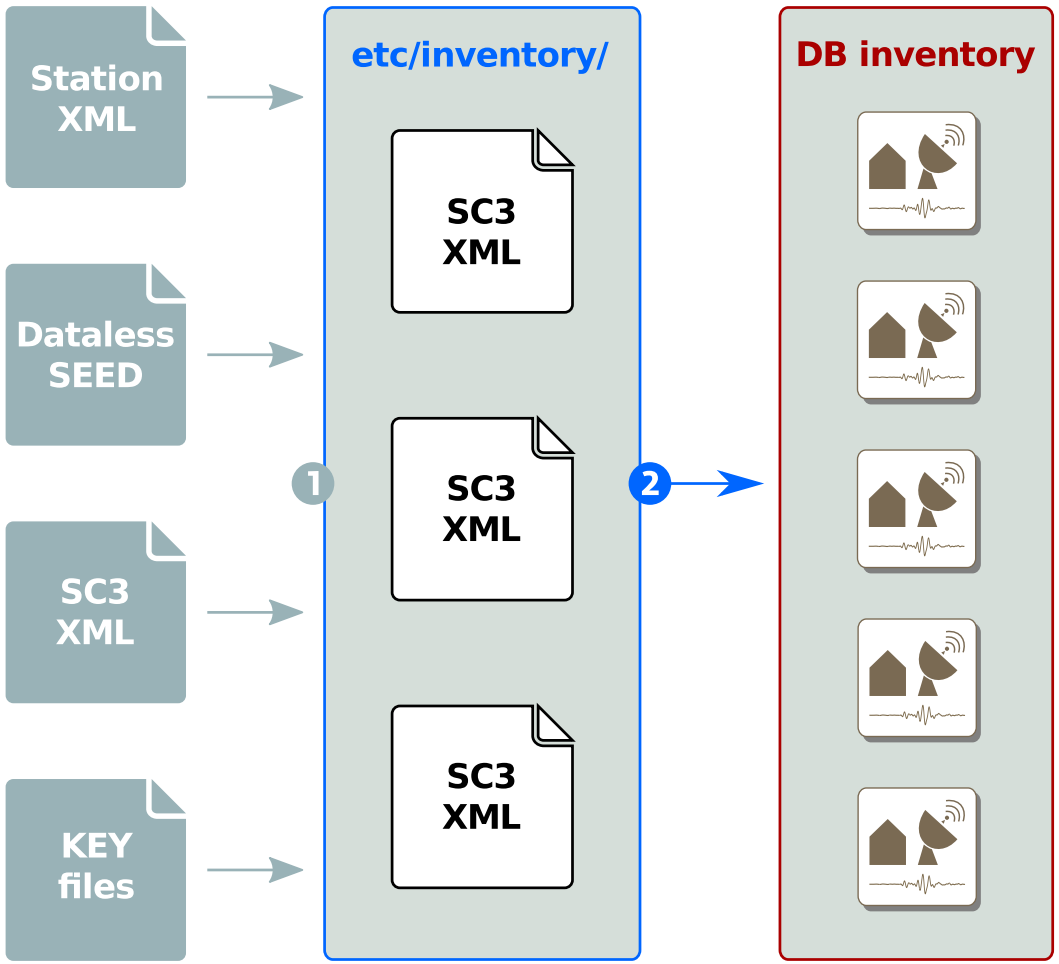
- Only one authoritative tool is allowed to update the SQL inventory database, in particular *scinv*
- It synchronizes inventory information in SC3 XML format exclusively
- External formats (current key files, dataless SEED, StationXML, ...) can be converted to SC3 XML by either SeisComP3 or external tools
- Users can review conversion results and even edit XML manually **in advance** to database updates
- one-to-one relationship between database information and inventory directory
- No error prone import_*, write config and sync_* workflows anymore
  $\rightarrow$ one sync channel only
- Cloning inventory information from another machine is as easy as *scxmldump* + *seiscomp update-config inventory*

Metadata aka Inventory - supported formats

gempa

- Arclink
- StationXML
- Dataless SEED
- Key files (SeisComP3 Zurich)
- Nettabs
- Nettabs2

Workflow



1. Convert external sources to SC3 XML and store it in etc/inventory/
2. Merge all inventory files in etc/inventory/, check for conflicts and synchronize with database (*seiscomp update-config inventory*)

Configuration

Configuration files

gempa

- Simple text files where each line is a name-value pair
- **WARNING**: In contrast to prior versions the parameter names are case-sensitive!
- Everything following an un-escaped '#' (hash) is a comment and ignored
- Blank lines and white spaces are ignored by the parser as well unless quoted or escaped
- Later assignments overwrite earlier ones so the order of lines is important

```
skyColor = yellow  # This is a comment

# The preceding empty line is ignored and previous setting "yellow"
# is replaced by "blue":
skyColor = blue
```

Configuration files (cont.)

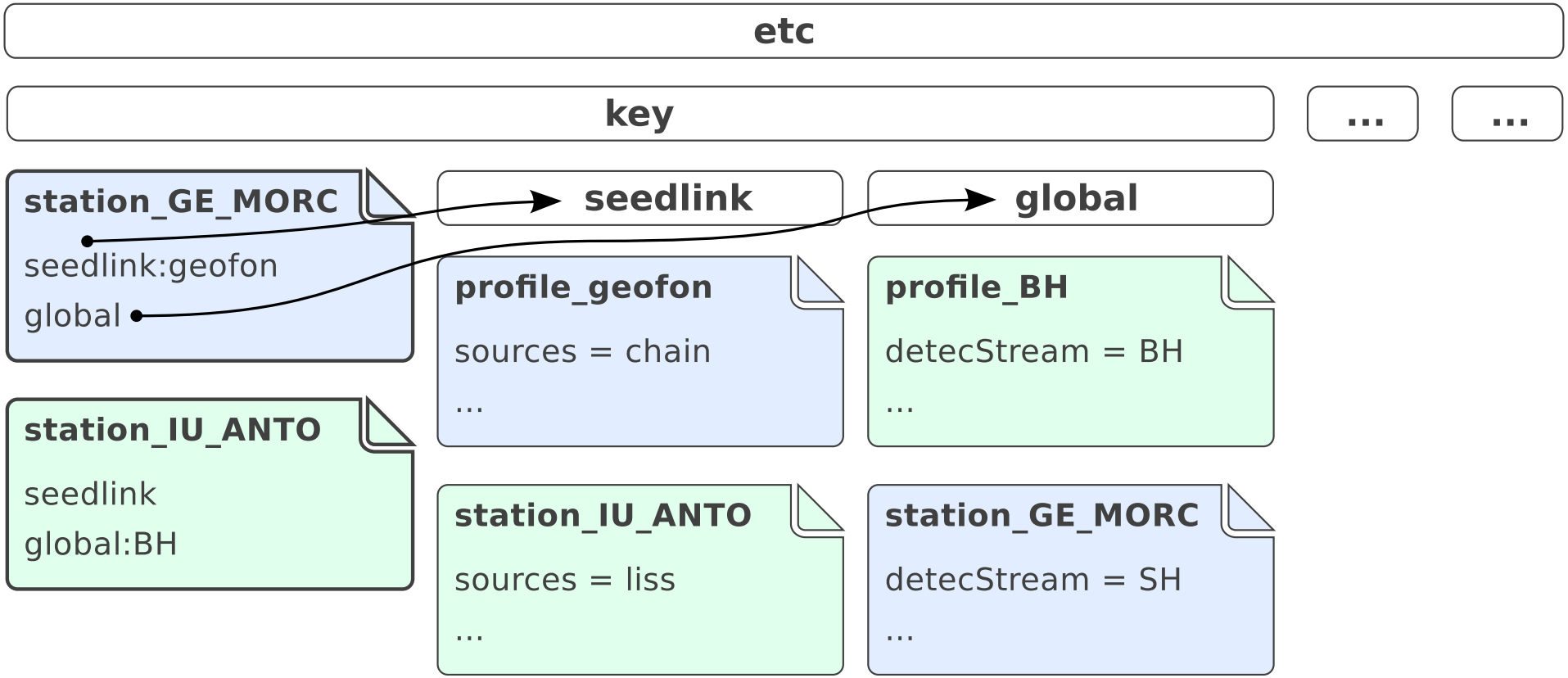
gempa

- Values can be either scalar values or lists where items are separated by commas
- If a value needs to include a comma, white space or any other interpretable character it can either be escaped with backslash (\) or quoted using double quotes (")
- Environment or preceding configuration variables can be used with ${var}

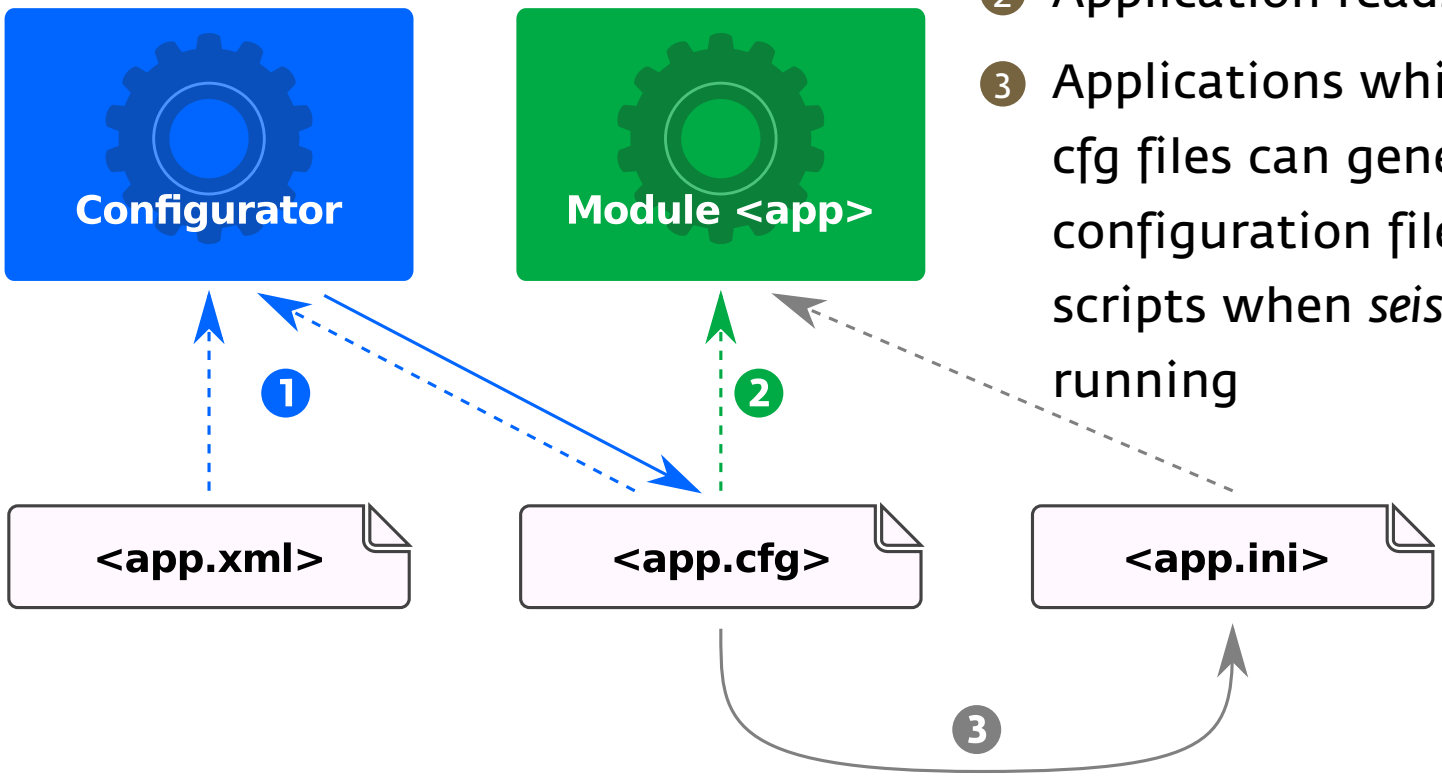
```
# This is a list definition
rainbowColors = red, orange, yellow, green, blue, indigo

# Append a value having a space and a comma to an existing list
rainbowColors = ${rainbowColors}, "violet, not purple"
# escaped values
tuples = 1\,2, 3\,4
# vs quoted values
tuples = "1,2", "3,4"
```

Key file structure

gempa



Entry point is a station key file (right under etc/key). Without it, bindings for a particular station are not used at all even if binding files exist.

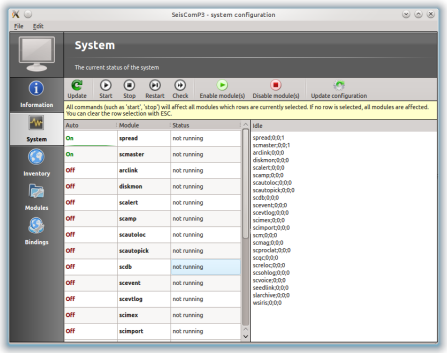


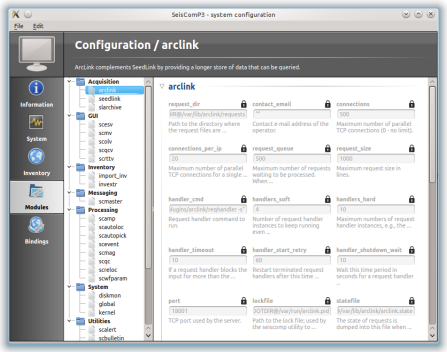
1. Configurator reads XML + cfg and writes back cfg
2. Application reads cfg directly
3. Applications which do not support cfg files can generate their native configuration files through their init scripts when *seiscomp update-config* is running

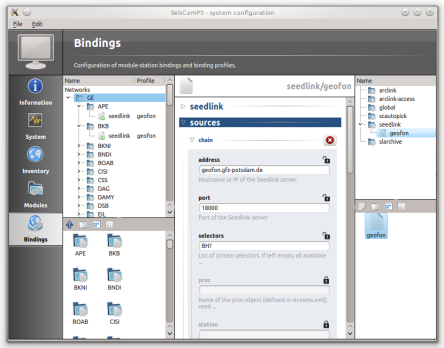
Configurator

gempa

- *scconfig* is a GUI driven configurator and system manager for SeisComP3
- Frontend to *seiscomp* process management (start, stop, ...)
- Supports to edit module configurations and to create/edit/remove bindings
- Imports inventory from various sources
- Operates in system (etc/) or user (~/.seiscomp3/) mode







seiscomp

gempa

- Central module manager
- Lives in $SEISCOMP_ROOT/bin/
- Fully written in Python
- Does not need a SeisComP3 shell environment to run → multiple instances can be managed within one shell session

```
sysop@host:~$ $ seiscomp3/bin/seiscomp stop
...
sysop@host:~$ $ seiscomp3.test/bin/seiscomp start
...
```

- Build-in help

```
sysop@host:~$ seiscomp3/bin/seiscomp help
Available commands:
 install-deps
 ...
```

gempa

seiscomp install-deps

- Installs OS dependencies to run SeisComP3. This requires either a 'sudo' or root account. Additional packages must be given as arguments.
- Example: *seiscomp install-deps base mysql-server*

seiscomp setup

- Initializes all modules and creates default configurations, e.g. SDS path

seiscomp enable|disable [mod1 mod2 ...]

- Enables/disables a module for *seiscomp start|stop*
- Creates/deletes status file *etc/init/modname**.auto***

seiscomp start|stop|restart [mod1 mod2 ...]

- Starts all enabled modules or a list of modules
- Stops all started modules or a list of modules

seiscomp check [mod1 mod2 ...]

- Checks if either all started modules or a list of modules are still running and starts them otherwise

seiscomp status [mod1 mod2 ...]

- Prints the status of all or a list of modules
- Gives warnings if a module should run but doesn't

seiscomp exec [cmd ...]

- Executes a command under the SeisComP3 environment
- Wrapper to launch SeisComP3 applications without the need to source the environment in advance
- Good for buggy desktop managers

seiscomp list [modules|enabled|disabled]

- Prints a list of all available, enabled or disabled modules

seiscomp update-config

- Updates module configurations and database
- Replaces *seiscomp config -> Write*
- Can be applied to a subset of modules

seiscomp print env

- Replaces the current *lib/env.sh* script

Source SeisComP3 environment

```
sysop@host:~$ eval $(seiscomp3/bin/seiscomp print env)
```

seiscomp print crontab

- Replaces *seiscomp print_crontab*

seiscomp alias

- Creates and removes aliases to modules, because a simple symlink to an existing binary is not enough anymore.

seiscomp help

- Prints all available commands or help for a particular command.
- Example: *seiscomp help alias*

Use cases

Initial setup

gempa

```
sysop@host:~$ seiscomp setup
============================================================================
SeisComP setup
============================================================================

This initializes the configuration of your installation.
...
options of your setup but helps to setup initial standard values.

----------------------------------------------------------------------------
Hint: Entered values starting with a dot (.) are handled
...
----------------------------------------------------------------------------

Agency ID []: _
```

Using the shell to assign profiles

gempa

```
sysop@host:~$ seiscomp shell
================================================================================
SeisComP shell
================================================================================
...
$ set profile seedlink chain GE.*
OK, 1 files modified
$ print station GE.MORC
[seedlink]
/home/sysop/seiscomp3/etc/key/seedlink/profile_chain
--------------------------------------------------------------------------------
sources = chain
sources.chain.address = geofon.gfz-potsdam.de
sources.chain.port = 18000
sources.chain.selectors = BH?
--------------------------------------------------------------------------------
```

Update module configuration

gempa

If configuration files or bindings were modified the module configuration needs an update. One option is to call *update-config* without parameters which will update the entire system and take care of the correct order of modules. There might be situations were only a particular module should update its configuration (testing, debug).

```
sysop@host:~$ seiscomp update-config seedlink
* starting kernel modules
starting spread
starting scmaster
* configure seedlink
+ network GE
  + station MORC MORC
    + source chain
```

Note: This is not necessary if the configuration file of a trunk module was changed.

Update binding configuration in database

gempa

To update the database configuration which is read by all trunk modules that support bindings, a trunk meta module exists which cannot be started but which can be configured. It takes all the configured bindings from etc/key and populates the database.

```
sysop@host:~$ seiscomp update-config trunk
* starting kernel modules
spread is already running
scmaster is already running
* configure trunk
+ default
  + read 139 stations
+ scautopick
  + read 139 stations
- database is already up-to-date
```

Update inventory in database

gempa

If new inventory files were imported which should be synchronized with the database, an inventory meta module exists which can be used along with *update-config*.

```
sysop@host:~$ seiscomp update-config inventory
* starting kernel modules
spread is already running
scmaster is already running
* configure inventory
WARNING: etc/inventory/README ignored: wrong extension
Parsing etc/inventory/inventory.xml ... done
Parsing etc/inventory/seiscomp3.xml ... done
Merging inventory ... done
Synchronising inventory ... done
Removing remaining objects ... done
Inventory is synchronised already, nothing to do
```

Setup system for autostart

gempa

```
sysop@host:~$ seiscomp enable seedlink scautopick scautoloc scamp scmag scevent
enabled seedlink
enabled scautopick
enabled scautoloc
enabled scamp
enabled scmag
enabled scevent
sysop@host:~$ seiscomp start
starting spread
starting scmaster
starting scamp
starting scautoloc
starting scautopick
starting scevent
starting scmag
starting seedlink
maximum number of open files set to 4096
```

Manage module aliases

gempa

Creating aliases is not as easy as creating a symlink to a module binary as in prior versions. Because of the new configuration framework and its requirements some more files need a copy. The user just needs to know that there exists a command: alias create/remove.

```
sysop@host:~$ seiscomp alias create scautopick2 scautopick
Copy default configuration: etc/defaults/scautopick.cfg ->
etc/defaults/scautopick2.cfg
Create app symlink: scautopick -> bin/scautopick2
Copy init script: etc/init/scautopick.py -> etc/init/scautopick2.py
sysop@host:~$ seiscomp list aliases
scautopick2 -> scautopick
sysop@host:~$ seiscomp alias remove scautopick2
Remove default configuration: etc/defaults/scautopick2.cfg
Remove app symlink: bin/scautopick2
Remove init script: etc/init/scautopick2.py
```

Machine readable status query

gempa

If one wants to write a monitoring script that needs to know the current status of all modules it would be hard and not future proof to parse the "is running" and "is not running" messages of *seiscomp status*. A csv (comma separated values) formatter is available in *seiscomp* which prints the result of the query in a machine readable format (*name;running;should-run;enabled*) along with some additional information.

```
sysop@host:~$ seiscomp --csv status
spread;1;1;1
scmaster;1;1;1
arclink;0;0;0
diskmon;0;0;0
scalert;0;0;0
scamp;0;0;1
scautoloc;0;0;1
scautopick;0;1;1
...
```

Check configuration

gempa

SeisComP3 comes with *scchcfg* which helps to detect problems such as syntax errors or conflicts with respect to case-sensitivity.

```
sysop@host:~$ scchkcfg scautopick
Read configuration files OK
Conflict #1
module.trunk.global.amplitudes.mB.signalEnd /home/sysop/.se...p3/scautopick.cfg:1
module.trunk.global.amplitudes.mb.signalEnd /home/sysop/.se...p3/scautopick.cfg:2
1 conflict detected
```

Sometimes it is hard to know where a configuration parameter is actually defined. *scdumpcfg* helps in this regard.

```
sysop@host:~$ scdumpcfg scautopick -P module.trunk.global.amplitudes.mB.signalEnd
module.trunk.global.amplitudes.mB.signalEnd
  content: 60
  source: /home/sysop/.seiscomp3/scautopick.cfg
```

gempa

scdumpcfg can also be used to to dump all bindings or to check a database binding parameter of a particular module.

```
sysop@host:~$ scdumpcfg scautopick -B
+ GE.MORC
amplitudes.enableResponses: false
detecFilter: RMHP(10)>>ITAPER(30)>>BW(4,0.7,2)>>STALTA(2,80)
detecStream: BH
timeCorr: -0.8
trigOff: 1
trigOn: 3
sysop@host:~$ scdumpcfg scautopick -B -P detecStream
+ GE.MORC
detecStream: BH
sysop@host:~$ scdumpcfg global -B -P detecStream
+ GE.MORC
detecStream: BH
```

Secondary acquisition server

gempa

Primary acquisition server collects data from external servers and the secondary server serves data for internal processing collected from primary server.

```
#!/bin/sh
# Remove all station keys
rm -rf ~/seiscomp3/etc/key/station_*_*
# Copy keys from primary server
scp -r user@primary:seiscomp3/etc/key/station_*_* ~/seiscomp3/etc/key
for i in `find ~/seiscomp3/etc/key -name "station_*_*"`; do
  # Remove active seedlink line and replace it by "seedlink:primary"
  sed -i "s/^seedlink[^#\ ]*/seedlink:primary/g" $i
done
# Update Seedlink configuration
~/seiscomp3/bin/seiscomp update-config seedlink
# Restart Seedlink
~/seiscomp3/bin/seiscomp restart seedlink
```

gempa

Setup Demonstration With *scconfig* ...