Snow agents
Simple Network of Work Agents

Mission specification

Revision: 0.14

Authors:
Johan Gustav Bellika

Revision history:

Issue	Details	Who	Date
0.14	Removed checkpoint, interactive, submissions, localNetw, remoteNetw, changeTargets, throwaway, returnTmpRes, returnAsap, exec_prior, pid, filePerm, restore, storage, policy_violations. The exec_prior is to be replaced by role based priority. Removed type attribute in result and luggage tags.	L Ilebrekke	02 Oct. 2008
0.13	Changed specification of result tag and added description of result_file tag.	JG Bellika	12 Aug 2004
0.12	Changed specification of result tag.	JG Bellika	23 June 2004
0.11	Made minor modifications to several tags and attributes.	JG Bellika	10 Feb 2004
0.10	Added maxTimeDelta and timeDelta tags to enable agreement on timeout values for requests.	JG Bellika	10 Feb 2004
0.9	Added McJid attribute for storing intermediate Mission controller address to agent tag.	JG Bellika	26 Nov 2003
0.8	Added mainMcJid attribute to agent tag.	JG Bellika	20 Oct 2003
0.7	Added specification of how hosting policy validation is stored in the mission specification during mission negotiation.	JG Bellika	8 Oct 2003
0.6	Reformulated specification of static and dynamic mission requirement and how hosting policies for each agent type is enforced.	JG Bellika	1 Oct 2003
0.5	Added user certificate to the agent tag in the mission specification.	JG Bellika	30 Sept 2003
0.4	Changed the specification of how and where legal values for optional mission requirements are expressed and stored.	JG Bellika	10 Sept 2003
0.3	Added activation and migration time limits.	JG Bellika	5 Aug 2003
0.2	Revised mission specification after definition of migration protocol.	JG Bellika	3 Aug 2003
0.1	Initial version	JG Bellika	9 July 2003

1 Introduction

The static mission specification or configuration information for an agent is loaded from the agent configuration file when an agent is instantiated on a host. In addition dynamic and/or optional mission specification is received from the agent daemon after authentication. The dynamic and/or optional mission specification is received / negotiated with a remote mission controller.

Both local (static) agent configuration and dynamic and/or optional mission specification is expressed in XML. The agent daemon enforces its policy regarding legal behaviour and legal mission specification by validating the specification using local XML schemas. The local XML schemas express the local policies for each agent type supported by the local agent daemon.

The XML tags in the mission specification describe resource needs and behaviour that is considered necessary to perform and control the agent mission. The mission specification is also a contract about resource usage between the mission controller, the agent daemon and the agent that all parties must obey. If not the behaviour of the system may become unpredictable.

The final specification for each agent in the mission is the result of a potential negotiation between a mission controller and an agent daemon on a remote host. Deviation from the overall mission specification is stored in a separate section in the mission specification that is reserved for each host.

There exists a ***MissionSpec.xsd*** schema within the development project that describes this document more formally. When updating this document or the schema make sure that they are in synch.

1.1 Static requirements

The agent daemon has a list of supported agent types. Which agent types that is supported and which user that is allowed to run each agent type is stored in an access control type of configuration file. This configuration file is read by the agent daemon and used each time it receives a mission (migrate or host) request. The file is maintained by the system administrator responsible for the host running the agent daemon. If an agent type is not supported, the mission is rejected immediately by the agent daemon.

The agent daemon knows the static requirements of a supported agent type because its behaviour and configuration is known before instantiation. Hence mobile code in binary or source code format is not possible within this system.

All agent types have a common base of possible behaviour. This behaviour may be the ability to migrate, clone, initiate sub missions, interact with other agents and users, respond to mission administrative commands, etc. Which behaviour that is supported or allowed by each agent type is expressed using a local configuration file (in XML format) and XML schema's which express the constraints on the agent type's behaviour. The local agent configuration contains a specification of the static requirements or restrictions on the agent type. The XML

schema expresses the policies for dynamic and/or optional behaviour for the agent type. The policies for the agent type are static by nature.

1.2 Dynamic and optional requirements

In addition to the static requirements for the mission, each agent type may have dynamic requirement such as storage requirements, computation time requirements, response time requirements or computation priority that may not be acceptable for the agent daemon. These requirements are negotiated together with the optional requirements before an agent may compute under the control of the agent daemon.

The hosting policies expressed in the XML schemas for each agent type may use default values for its preferences. If the mission specification does not contain a value for an attribute, the default value is used. It may use fixed values for absolute requirements for attribute contents. The XML schemas may use a list of enumerations to express lists of possible attribute values or type specification to enforce that the specification is within legal boundaries for the system.

For resource requirements like storage, computation priority, response time and computation time, the agent daemon implements the resource usage preservation policy.

1.3 Precedence rules

The tags used in the mission specification may contain information that specify optional requirements or "nice to have" features for the agent mission. If the negotiated specification deviates from the overall mission specification, it is stored below the appropriate host tag in the mission specification. See below.

The local agent configuration file contains tags that specify the static requirements. If the overall mission specification contains the same type of specification as the negotiated one, the negotiated specification takes precedence over the overall mission specification.

In summary, the mission specification stored under the host tag always rules. If the specification is not present under the host tag, the valid one is the overall mission specification. If the specification is not available in the overall mission specification, the default/fixed values given in the policy specification (XML schema) for the agent type apply. If the specification is not available in the agent configuration file, the default value given in this document applies.

2 Specification tags

Question: How can we be sure that the mission specification is correct and not changed underway?

2.1 The "agent" tag

The agent tag contains administrative information about the agent.

- "missionMode": Possible values "jump" or "spread"

- "type": Contains the application type. The type must be supported by the receiving host.
- "version": The version of the mission type.
- "name": Name of the mission. Used as name when browsed.
- "idempotent": Values 'yes' or 'no'. Is it possible to rerun the agent mission to produce the same mission result?
- "requesterjid": Contains the jid of the mission requester. It implicitly defines the host for the agent mission and origin mission controller. Eks. `requesterjid='WHOUser@dipato.nhn.no/exodus'`
- "mainMcJid": Contains the jid of the main mission controller. The value of this attribute may change during a mission as a result of a mission handover between two mission controllers. See the mission handover protocol in the Interaction Protocols specification. The value of this attribute is set by the main mission controller. Eks. `mainMcJid='mc.dipato.nhn.no'`
- "user": Contains the name of the user known by the jabber daemon. Must also be known by the operating system on each host where this user can perform agent missions. Password of the operating system should (must) be different from the jabber password (if used).
- "user_certificate": Contains a certificate proving the identity of the user.
- "birthid": Birth id of mission agent. Originally created by the mission controller. The value is a SHA hash of the requesters jid and the creation time for the mission. The birthid is the key to the mission controller's mission table. It is also used as identifier for communication between mission controller and agent daemon.
- "McJid": Contains the Jid for the intermediate mission controller providing the migration service to the agent. This attribute is set by the mission controller providing migration service. The attribute has no meaning in spread mode missions.

2.2 The "mission" tag

The mission tag may contain the following attributes. If the attribute don't exists, default values are assumed.

- "public": Default attribute value "no". Is the presence of the agent public, ie. may it be browsed by others than the agent daemon, mission controller and the requester?
- "descendantsCount": Default value "1". Number of allowed agent generations. If the mission agent starts sub missions this value is decremented in the mission specification of the sub mission used to negotiate the mission. The default value does not allow sub mission.

2.3 The "maxTimeDelta" tag

Because host may go offline, be online in periods, long response times must be possible. This tag contains the value of the maximum allowed delay between when a message is sent and when the message is received. Each mission controller and agent daemon is configured with a value called "pollDelay". This value states the delay caused by polling for messages to a jabber server within the network that stores messages when the Snow Agent server is offline. The pollDelay corresponds to the maxTimeDelta value for each daemon. If the daemon has a pollDelay value greater than the value of the maxTimeDelta tag in the mission specification, then the daemon cannot participate in the mission and must deny service. A mission controller can therefore assume that an initial request has failed when time > 2 X the value of maxTimeDelta has passed.

This tag has a fixed value. See [1] for details. The tag has the following attributes:

- "unit": May take the values 'seconds', 'minutes', 'hours', 'days'
- "value": Contains the value of the duration.

2.4 The "timeDelta" tag

The value of this tag must be set to the maximum value of the two daemons negotiating a visit. When set, and both participants have agreed, then the sender of a message can assume that a request have failed when time equals 2 X the value of the timeDelta tag has passed. See [1] for details.

The mission controller sending a migrate or host request to an agent daemon must set the value of this tag to its own value for pollDelay. The agent daemon receiving a mission specification must check this value and replace the value if its pollDelay value is greater than the mission controller requesting service. The mission controller always has to accept the agent daemons value for timeDelta, as long as it is below the maxTimeDelta value.

The tag has the following attributes:

- "unit": May take the values 'seconds', 'minutes', 'hours', 'days'
- "value": Contains the value of the duration.

2.5 The "missionDuration" tag

If this tag is specified the mission controller accepts to keep information about the mission in a working directory with the same name as the mission birthid, until the requester removes it, or the remove timestamp ("remove_ts") is reached. If the tag is not specified, the mission is removed immediately after termination, failure or successful completion. If the mission is not finished within this deadline, the mission is simply removed from the mission table of the main mission controller. The running agent is required to terminate when the mission duration deadline is reached without informing the main mission controller or waiting for acknowledgement from the main mission controller.

The tag may have the following attributes:

- "remove_ts": Deadline for removing the working directory of the mission. The timestamp is specified using the ISO 8601 standard notation YYYYMMDDTHH:MM:SS.

- "infinite": Default value is "no". "Yes" if there is no mission duration deadline. If "yes", the "remove_ts" attribute is ignored. The mission is kept in the mission table until it finishes, fails or is terminated by the mission requester. The mission storage is kept until explicitly removed by the requester. NOTE: *This option must be used with caution because the mission controller may risk running out of disk storage.*

2.6 The "visitDuration" tag

This tag contains the value of the maximum allowed duration of a host visit. The tag has the following attributes:

- "unit": May take the values 'seconds', 'minutes', 'hours', 'days'
- "value": Contains the value of the duration.
- "ts_start": Timestamp for start of visit at current host using the ISO 8601 standard notation YYYYMMDDTHH:MM:SS. Set by the agent on startup.

Eks. <visitDuration unit=hours' value='25' ts_start='20030616T20:56:40'/>

Note: In spread mode the deadline for the visit should not exceed the deadline set by the missionDuration tag.

2.7 The "negotiationLimit" tag

This tag contains the value of the maximum allowed duration of negotiating a host visit. Because participating host may go offline, be online in periods, a long negotiation limit must be possible. This tag makes it possible to have very long negotiation time limits. Because the negotiation time is a part of the maximum visit time (set by the 'visistDuration' tag), the time value of this field must be less than the visit duration time. The time value of this tag must also be greater than at least 2 X the value of the maxTimeDelta tag.

The tag has the following attributes:

- "unit": May take the values 'seconds', 'minutes', 'hours', 'days'
- "value": Contains the value of the duration.
- "ts_start": Timestamp for start of negotiation using the ISO 8601 standard notation YYYYMMDDTHH:MM:SS. Set by the mission controller issuing the request. Eks. <negotiationLimit unit='hours' value='24' ts_start='20030616T20:56:44Z'/>

2.8 The "activationLimit" tag

NB! Not implemented in agent.

This tag is only needed when an external user model is used. See the luggage tag below. If the user model is internal this tag should not be used. The activation period is the time from instantiation of the agent, until it receives the user model and becomes fully operational. This tag contains the value of the maximum allowed duration of the activation period during a host visit. The tag has the following attributes:

- "unit": May take the values 'seconds', 'minutes', 'hours', 'days'
- "value": Contains the value of the duration.
- "ts_start": Timestamp for instantiation which marks the start of the activation phase using the ISO 8601 standard notation YYYYMMDDTHH:MM:SS. Set internally by the mission agent and sent as part of the status message to the main mission controller. Eks. <activationLimit unit='hours' value='24'/>

2.9 The "migrationLimit" tag

NB! Not implemented in mission agent

The migration time limit specifies the time period reserved for negotiation the visit with the next host in the itinerary. The time for starting the migration phase is defined by:

Start of visit + visit duration – migration time limit

If migration is not possible within the migration time period, the mission has failed. The tag has the following attributes:

- "unit": May take the values 'seconds', 'minutes', 'hours', 'days'
- "value": Contains the value of the duration.

If the tag is not specified the migration phase is started when the agents task is performed. In such cases the migration limit is defined as:

migration time limit = Visit duration – activation time – working time

Note: This tag has no meaning in spread mode missions.

2.10 The "targets" tag

The targets tag contains the list of target hosts that should be visited during an agent mission. The tag has one possible attribute.

- "deviations": Values 'yes' or 'no. Default value is 'no'. If 'yes': In spread mode, the mission controller stores information and the result of the mission negotiation that deviates from the overall mission specification, below each corresponding host tag. It is therefore possible to have differences between the hosts participating in a spread mode mission. If 'no', deviation between the hosts is not allowed and the mission is terminated

if one of the required hosts is not able to meet the mission specification.

2.11 The “host” tag

Mission specification that deviates from the overall mission specification is stored under the host tag. If there are conflicts between the overall mission specification and the version stored under the host tag, the version under the host tag take precedence over the overall mission specification.

The host tag may have the following attributes:

- “name”: The Fully Qualified Domain Name (FQDN) hostname for the target. DNS is used to lookup and route messages between hosts.
- “seq_num”: The attribute contains a sequence number > 0 if the targets must be visited in some predefined order. If this attribute is set then each host in the taget list may be contacted only once. (The ‘retries_left’ attribute below has no effect, if specified). Note: this attribute is not used in spread mode missions.
- “optional”: Possible values ‘yes’ or ‘no’. If host specified as optional=’yes’ it may be skipped without terminating the overall mission, if optional=’no’, the mission is discarded if the host cannot be visited.
- “status”: The status of an agent visiting a host may take the following values: ’ready’, ’working’, ‘visited’, ’failed’, ’denied’, ’unknown’. Hosts not tried or visited yet, have the status ‘unknown’.
- “retries_left”: Default value for this attribute is 0. If the targetlist is sequential then this attribute has no meaning. If a host does not respond to negotiation within the negotiation time, the host may be contacted again, from another host. For each failed negotiation the value is decremented until the value of this field is zero. Note: Each host is contacted only once from its current location, hence this attribute is not used in spread mode missions.
- “ts_start”: Timestamp for start of visit using the ISO 8601 standard notation YYYYMMDDTHH:MM:SS. Only applicable in jump mode and is set by the agent itself.
- “ts_stop”: Timestamp for stop of visit. Only applicable in jump mode and is set by the agent itself.

2.12 The “luggage” tag

The luggage tag is used by the agent to store mission data. The user model for the agent mission is an example of mission data stored below this tag. The user model may also be stored in files that are communicated directly between the mission controller (on behalf of the

requestor) and the mission agent, if allowed. References to such files are stored under this tag. If the user model is not allowed to be transferred outside the normal XMPP messages, they are transferred as base64 encoded messages. Each agent type must describe the requirements for the contents of this tag.

2.13 The “result” tag

The result tag may be used to store accumulated mission result if not reported directly to the mission requestor or someone else. It is basically up to the mission designer/implementer to decide the contents under this tag. The result tag may for example be populated with tags for each host that reports result to the mission controller. When the mission has finished the contents of the result tag may be sent to the mission requestor. Attributes:

- “recipient_jid”: Contains the JID to the recipient of mission results. The requester, Mc or somebody else.

2.14 The “result_file” tag

Mission results stored in separate files are transferred between entities in base64 encoded in-band byte streams. The specification of the file is sent to the receiving entity using a “set status visited”, “set status finished” or a “set result” message. The result_file tag is placed as a child tag to the result tag specified above. The tag contains the following attributes:

- “filename”: Contains the name of file transferred.
- “hash”: MD5 hash value of the file.
- “date”: Date of the file on format YYYYMMDDHHMMSS.
- “desc”: Contains a description of the file transferred.
- “size”: Size of file in bytes.
- “SID”: Session id for transfer of file.

2.15 The “debug” tag

This tag is used to debug a single mission of the agent type. If specified the mission agent produce debug messages that are routed through the jabber system to the specified debug message receiver. This option is purely intended for troubleshooting agent mission on problematic hosts and must be used with extreme caution. The tag is used by the agent daemon to instantiate the mission agent with the debugging switch on. The tag has the following attributes:

- “value”: Default value is ‘no’. To enable debug. Set this value to ‘yes’.
- “zonelist”: This attribute should contain the list of zones that are targeted for debug. Keep this list to a minimum.
- ‘receiver’: This attribute must contain the jid of a valid and online jabber user. Eks. <debug value='yes' zonelist='ma,mac' receiver='u1@dipato.nhn.no'/>

2.16 Example specification

Below you see an example mission specification while a mission agent is visiting the host nhn271.nhn.no. It was requested from 'WHOUser@dipato.nhn.no/exodus' to mc.dipato.nhn.no.

```
<spec>
    <agent   missionMode='jump'
             type='epidemiologyquery'
             version='1.0'
             name='WHO-Epidemiology'
             idempotent='yes'
             user='WHOUser'
             birthid='123abc'
             requesterjid='WHOUser@dipato.nhn.no/exodus'/>
    <mission public='no'
             descendantsCount='3'/>
    <visitDuration
             unit='hours'
             value='25'/>
    <negotiationLimit
             unit='hours'
             value='24'
             ts_start='20030616T20:56:44'/>
    <targets deviations='no'>
      <host  name='nhn271.nhn.no'
             seq_num='1'
             optional='no'
             status='visited'/>
      <host  name='nhn332.nhn.no'
             seq_num='2'
             optional='no'
             status='unknown'/>
    </targets>
    <luggage/>
    <debug value='no' zonelist='ma,mac' receiver='u1@dipato.nhn.no'/>
</spec>
```

2.17 Leftovers

3 References

[1] Lamport, L. Using Time Instead of Timeout for Fault-Tolerant Distributed Systems. ACM Transactions on Programming Languages and Systems, Vol. 6, No. 2 April 1984, 254-280.