A solution for autonomic software self-management

Authors: E. Troch, C. Herssens, E. Scotto – Siemens AG

1 Introduction

1.1 Motivation to introduce software self management

Software management in existing telecommunication networks is used for:

- Software upgrade: new or modified functionality are added to the Network Element (NE) by installing new software releases
- Software correction: the basic functionality is not changed. It is usually by means of patching or by limited substitution or addition of software modules.

In this paper the term "upgrade" refers, unless explicitly stated, to both used cases.

Software upgrades are a significant part of the OPErational eXpenses (OPEX) because of:

- personnel costs, mainly because highly qualified experts are needed
- effort, because of the necessary manual steps to be done before, during and after the upgrade

The costs obviously increase as long as the network complexity increases, in terms of number of NE's and frequency of release of new software versions or corrections.

Failed upgrades have an economic impact for the network operator, whose extent depends on the severity and duration of the outage, as well as on the type of affected NE and its role in the network topology.

The self management addresses the above issues by taking over and automating manual tasks from the operator.

1.2 Requirements on software self management

The main requirements to self management can be grouped in two areas:

- Automatic execution: the NE runs the necessary preparation tasks, the upgrade itself as well as post upgrade checks. In case of failures of any kind during or after the upgrade, the NE starts a recovery procedure and the previous status is restored without outages
- Autonomic management: the NE makes sure on its own that the most recent software version runs, by defining when, what and how software shall be updated. The actions to perform the upgrade are dynamically assembled by the NE itself and, if necessary, synchronized with other NE's

Other requirements and issues to be addressed to implement a commercial solution for software self-management are:

- the solution must be general enough to be applicable for (ideally) any NE in a network, hiding the NE specifics for the operator staff
- automatic distribution of released software from manufacturer to network operators
- autonomous software uploading from 3rd party portals

- autonomous installation of software for new installed hardware (plug and play)
- scalable solution for large networks

1.3 Migration to self-management

The migration steps to self management will most likely be in the following sequence:

- Automation: preparation, execution and post upgrade checks can be automated up to a certain degree
- Autonomic management: move the decision and the planning of an upgrade from operator to the NE's

Though in the IT world some degree of automation and autonomous software management represents the state of the art, just reuse of IT technology will be not sufficient because specific requirements of the telecommunication world must be also fulfilled, as for example:

- High reliability: for example a NE must at least fall back in the situation before the software management action, regardless of the kind of error occurs
- Synchronization: in many cases dependencies among NE's imply the need to plan the upgrade execution for each NE in a certain sequence and/or to run an upgrade step depending on the outcome of the previous step
- Interactivity with O&M craft, for example to control the upgrade steps

In the long term, however, in order to implement a full autonomous software management, input from basic research will be needed as many issues are today still open.

2 Proposed solution for self-management

2.1 Basic solution

The proposed basic solution covers the functionality required for the "full automation" and "semi-autonomic" SW management.

The coordination model is logically centralized and physically distributed:

- A central software management server on the Element Manager distributes a main workflow to local software management agents on NE's
- The agents customize and run the workflow under consideration of the local configuration data, software status and policies set by the network operator

This allows semi-autonomic management of the NE software. The same scheme is used for the agent themselves:

- A central agent deployment server on the Element Manager holds the most updated agent software
- An agent framework allows self management of the agent software on the NE and keeps them updated and running

The semi-autonomic process in both software management agent and agent framework is based on XML files, describing the workflow and providing configuration data.

The mentioned key entities for the implementation of the semi-autonomic management (see fig. 1) are described in more detail in the following paragraphs.

The full automation implies the automation of preparatory and post-upgrade tasks, such as checking hardware status, disk filling grade etc. Some of such tasks depend on product specific requirements. In theory, they can be hidden to the EM by leaving the customization of the workflow to the agents.

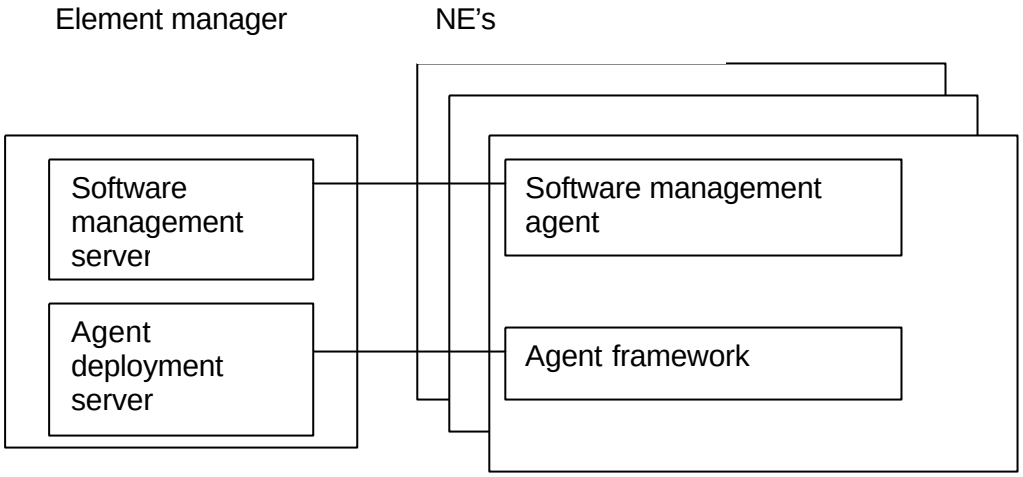


Fig 1: Key entities for the implementation of the semi-autonomic management

2.1.1 The Software management server and agent

When the software management server requests the agent to perform a SW management task, the main workflow and the control files are downloaded to the agent.

The *main workflow* is customized on the basis of a number of prepared XML control files. These control files are the *abstract state machine* description for the update type to be executed, as well as a set of configuration data, specific of the NE.

The output of this process is a concrete state machine consisting of actions and related states. An action can be:

- a command, for example: put disks in split mode, process restarting
- an user input, for example: requesting confirmation to start a process

As necessary, software packages are downloaded from the server on the EM by means of common file transfer protocols like FTP.

After executing an action, processing continues in follow-up states depending on the return code of the action and the local software status is updated.

The order of the upgrades among NE's is distributed in advance by the EM. The actual solutions to implement synchronization among NE's have to be further investigated and they are subject to research activity.

2.1.2 Agent deployment framework

In order to allow management, upgrade and operation of the agents, an agent deployment server runs on the EM and an agent framework runs on the NE's. The framework provides common functions to different type of agents on the NE.

The framework makes sure that the agents are started upon NE startup, according to the agent configuration data. The framework also polls periodically the server to check for new agents or new agent software versions.
The communication with the agent deployment server is possible via JNLP protocol, embedded in HTTP messages.

The data for the agent configuration, as for example the software management server address, are stored on NE in MIB's.

2.2 Extended solution

In order to have a more scalable solution, the agents on a NE may overtake also the role of agents for a cluster of NE's of the same type. In this way the EM has a more abstract view of the NE's (i.e. as cluster) and the overall performance is improved.

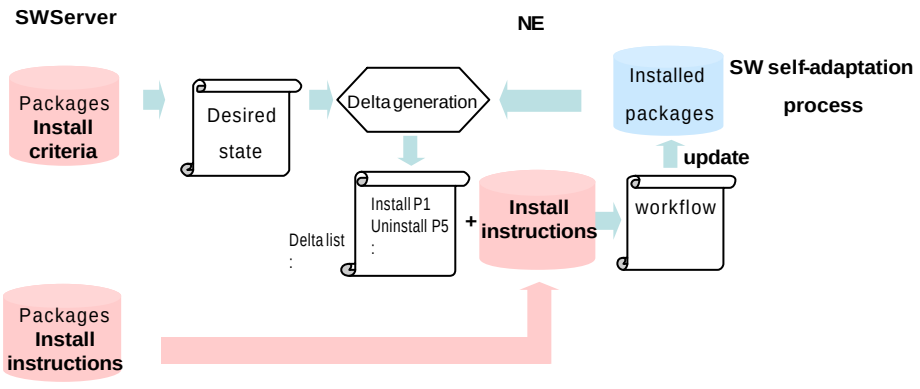
Further possible extensions of the basic solution, described in more detail below, are:

- dynamic assembly of desired software state, comparison with the actual state and generation of an upgrade workflow
- the introduction of an optimized overall time schedule for the upgrades of each NE

The long term vision extends the proposed solution over multiple domains and applies to OEM products as well.

2.2.1 Delta software supervision

For realising autonomous SW management, the SW agent can deploy the paradigm of "**delta software supervision**" for deciding when, what and how SW updating shall be performed. This mechanism assembles first a list of SW packages that are expected on the NE (desired state). The list and the sw itself are stored in a repository in the EM.
Next a "delta generation" function compares the desired state with the installed packages on the NE (actual state).
After the delta generation, the SW agent dynamically assembles the workflow by concatenating the install instructions provided with the software packages. Finally, the status of the installed software is updated according to the result of the upgrade actions.



2.2.2 Optimized time schedule

The workflow customization done at NE level cannot take into account some network level policy that the operators may wish to enforce; as to implement it knowledge and control of the entire network would be necessary in the NE.

For this purpose, the NE adds the time information in the customized workflow, which is derived from the local configuration information as well as from the downloaded install instructions. The customized workflow is not executed but sent to the EM as a "proposal". The EM collects the proposal from every NE impacted by the upgrade and applies the network level policies. The optimized workflow is finally sent back to the NE's which than run it.

3 Conclusions

By moving the execution of the SW management workflows away from the network management system, a very **scalable and generic solution** is realized.

Due to the autonomic decisions being taken by the SW agent and high automation, the effort for the O&M field personnel is significantly reduced. This results in **less OPEX costs** and less danger for failed upgrades. Additionally, it is possible to release new features more frequently, resulting in a **faster time to the market**.

The reliability of the installed SW is also improved by the complete SW status verification before/after each update and by the periodic SW integrity audits.

Self-management can be deployed for almost any task SW management task: first installations, new SW releases, version updates and patches.

First steps to self management can be adopted from the IT industry, but for a full autonomic sw management research in autonomic computing will be needed.

4 Abbreviations

EM	Element Manager
IT	Information Technology
MIB	Management
NE	Network Element
OPEX	Operational Expenses
OS	Operating System
SW	Software
XML	Extensible Markup Language