

# Completeness and consistency validation of IT configurations

whitepaper

## Executive Summary

This whitepaper introduces the technology of completeness and consistency validation of IT configurations. Basic concepts of completeness and consistency checking are introduced and the automated validation technology of OptXware is described.

## 1. Completeness and Consistency Checking

Completeness checking in the IT business is used both in the planning phase and during operation in order to assess the IT infrastructure from various aspects in order to make sure that the system meets the expectations imposed on it. Such aspects can be: Functional conformance: whether the current configuration can deliver the service it is expected to deliver – this encompasses from the network level (wiring, router and firewall configurations, etc.) to the application server middleware level a checking a multitude of elements and their relationships.

- *Dependability properties*: whether the current configuration meets the expected level of dependability in terms of availability, reliability, fault tolerance, resilience and so on.
- *Security properties*: whether the current configuration ensures the expected CIA (confidentiality, integrity, availability) properties – taking into account both internal and external threats and vulnerabilities caused by misconfiguration or erroneous software components (the existence of which in most cases is also a configuration error – apart from zero-day attacks and relatively ‘new’ exploits, the threat can be eliminated by installing a corrected version of the software).
- *Performability properties*: whether the current configuration ensures the level of operation that is agreed on in Operational Level and Service Level Agreements (OLA-s and SLA-s).
- *Regulatory compliance*: whether the current configuration meets the conditions that are imposed on the IT operations of an organization in a given sector by regulatory bodies.

These activities are all validation tasks, where the conformance of the system to a certain specification is ascertained. Due to the broad coverage of the term ‘configuration validation’, the approaches and tools used certainly widely vary; the validation task can be the processing of a simple checklist – as for example checking that all company firewalls disallow incoming connections with the exceptions of the Demilitarized Zone (DMZ) – but can involve heavy mathematic modeling and analysis, too, for instance for the what-if performance analysis of the infrastructure under usual load patterns.

In our vision, completeness analysis shall be based on a Configuration Management Database (CMDB), since in the absence of configuration management solutions, the information about the system that the validation works with has to be collected manually and/or using ad-hoc automation solutions – both being error prone and costly approaches. From this point of view, CMDB-s that a) have an ‘all-encompassing view’ of various configuration aspects, b) are maintained automatically or semi-automatically and c) were designed to be a configuration information middleware play a huge enabling role.

CMDBs per definition contain a consistent view of all configuration aspects of a system. Thus, configuration validation tasks with a well-maintained CMDB can be done on the model contained in the CMDB. Today, no consistent solution exists for supporting CMDB-based validation. Those validation tasks range in complexity from checklist-type evaluations through metamodel-conformance checking (i.e., the constraining of the set of allowed configurations through visual modeling means,

instead of natural language or manually programmed checks) to transformation to the input of special purpose analysis tools. Accordingly, this field can be considered as a market gap. Also, while a CMDB is a 'global view' and as such is usually a big database, unallowed real-world changes should be acted upon as fast as possible. We are working on the platform prototyping and the supporting of on the fly algorithms allowing instant alarms upon such changes.

With ITIL version 3, the CMDB became a central concept in the IT Infrastructure Library. However, currently it is used at most as the central CI and CI relationship database that can be accessed during the steps of an ITIL process.

A CMDB is a near real-time model of the whole 'configuration space' of an enterprise. Accordingly, if configuration changes of the system are managed by ITIL processes then by analyzing system models extracted from the CMDB the ITIL configuration management processes can be validated for correct implementation. Also, as artifacts of ITIL processes should also be stored in the CMDB according to the standard, this is a way to reach a central platform for the evaluation of process metrics.

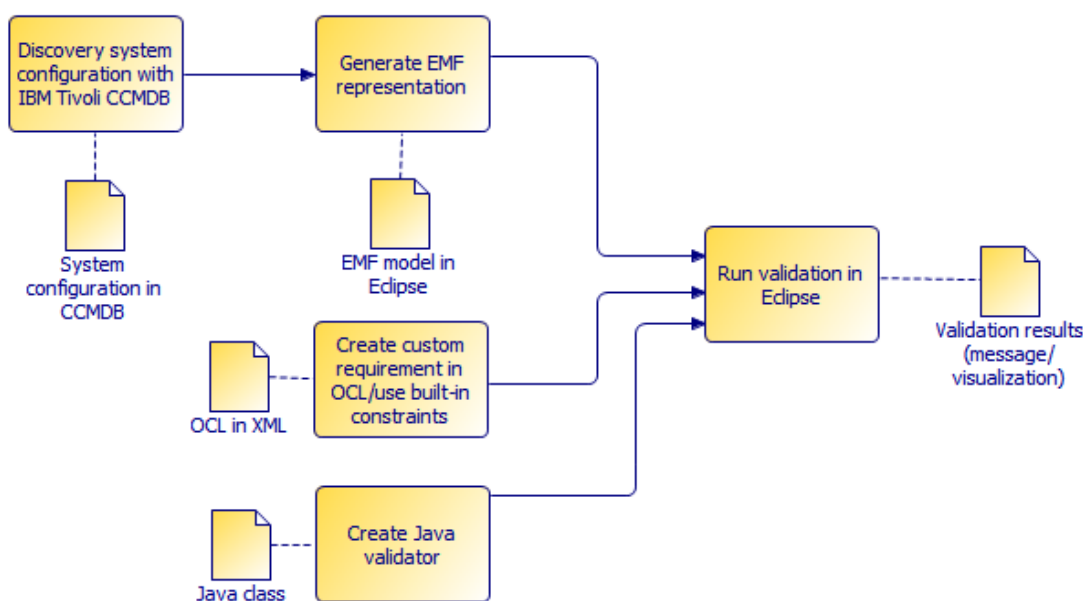
*"Customers who use our completeness checking services can be sure that the configuration of their IT system is free of ambiguities and unexpected functional and QoS properties. And this assurance is provided continuously..."*

## 2. OptXware technology

One of the leading tools in the field is the Change and Configuration Management Database (CCMDB) of the IBM Tivoli product line. Currently it is of strategic importance in the IBM service management strategy; on the one hand, it fulfills the role of configuration data middleware; on the other one, it also features an integrated process environment that allows users to implement and automate ITIL or ITIL-based IT service support processes on top of and tightly integrated with the CMDB.

OptXware offers an **Eclipse plugin** which processes configurations stored in CCMDB. Such configurations are first filtered to sort out relevant data. The Java API of the tool is used to gain information and build an Eclipse Modeling Framework (EMF) representation of the configuration. The quasi-standard Object Constraint Language (OCL) is part of the QVT specification of the Object Management Group, aiming at the definition of "rules" on software models. The plugin should support OCL queries, having built-in requirements and offering the possibility to define custom constraints in XML files.

Moreover, the plugin also offers the possibility to create Java validators; these are necessary to execute queries on the model (as OCL retrieves only locally available information via navigation steps). Such validators can perform complex queries on the EMF model, e.g. compare different parts of it, etc.



*Validation of IT configurations.*

Sample **OCL constraints** are the following:

"All AIX machines should contain at least 2 CPUs".

"All machines running ESX server should have CPUs of the same type."

"No netcat software should be installed on machines with publicly available IP address."

"A warning should be thrown if any Windows machines have IIS enabled."

"All machines should have different IP addresses."

"All MAC addresses should be different."

"No machines should have an UNKNOWN value for Audit State."

"The system partition of all Windows machines should have an available space of X MB."

"All operating systems should have at least 2 different DNS resolve entries."

The following constraints can be implemented by using dedicated **Java validators** due to current limitations of OCL evaluation on EMF models:

"All AIX machines should have at least 2 GB memory."

"All ESX servers should have at least 2 GB memory."

These requirements can be extended by the system engineer or by consulting with OptXware system management experts. Validation results in textual error messages and violation of requirements are also visualized by markings in the relevant part of the model. This way an intuitive yet precise feedback is available, supporting the system engineer to create and maintain correct and regulatory compliant IT configurations.

## Contacts

Imre Kocsis

Address: 1137 Budapest, Katona J. u. 39.

Phone: +36 20 514 6881/ Fax: +36 88 420273

Web: [www.optxware.com](http://www.optxware.com) / E-mail: [kocsis@optxware.com](mailto:kocsis@optxware.com)