RDBMS for CDB ?

Véronique Lefébure
IT-FIO/FS
Brainstorming, March 8th 2005

Outline

- CDB Requirements
  - Please read http://it-div-fio-lcg.web.cern.ch/it%2Ddiv%2Dfio%2Dlcg/qcancio/cdb/cdb_requirements_usesases.htm before this presentation, if possible.
  - Also doc about PAN at http://hep-proj-grid-fabric-config.web.cern.ch/hep-proj-grid-fabric-config/documents/pan-lisa.pdf
- An RDBMS to replace PAN ?
  - Features of PAN
  - What exactly would be replaced ?
  - Motivations:
    - Arguments in favour of PAN
    - Arguments in favour of an RDBMS
- A concrete example
  - the SW configuration: an RDBMS prototype
- Estimation of Manpower & Time needs **if** we go for an RDBMS

CDB Requirements: Summary

(See http://it-div-fio-lcg.web.cern.ch/it%2Ddiv%2Dfio%2Dlcg/gcancio/cdb/cdb_requirements_usesases.htm)

1. Content scalability
2. Grouping of data (for re-use)
3. Hierarchy
4. Inheritance (for no duplication of data)
5. Overwriting
6. Data types (basic, compound, user-defined)
7. Validation
8. Schema: evolution and fields optional/obligatory
9. Data transformation functions
10. Extensibility (for schema, data types, functions)
11. Consistency
12. Transactions
13. Rollback
14. History
15. CDB user scalability
16. Abstraction
17. Content portability (not a CERN requirement)
18. Data read access
19. Adding of new (sub) structures
20. Modification performance
21. Software availability and portability (not a CERN requirement)
22. User interfaces

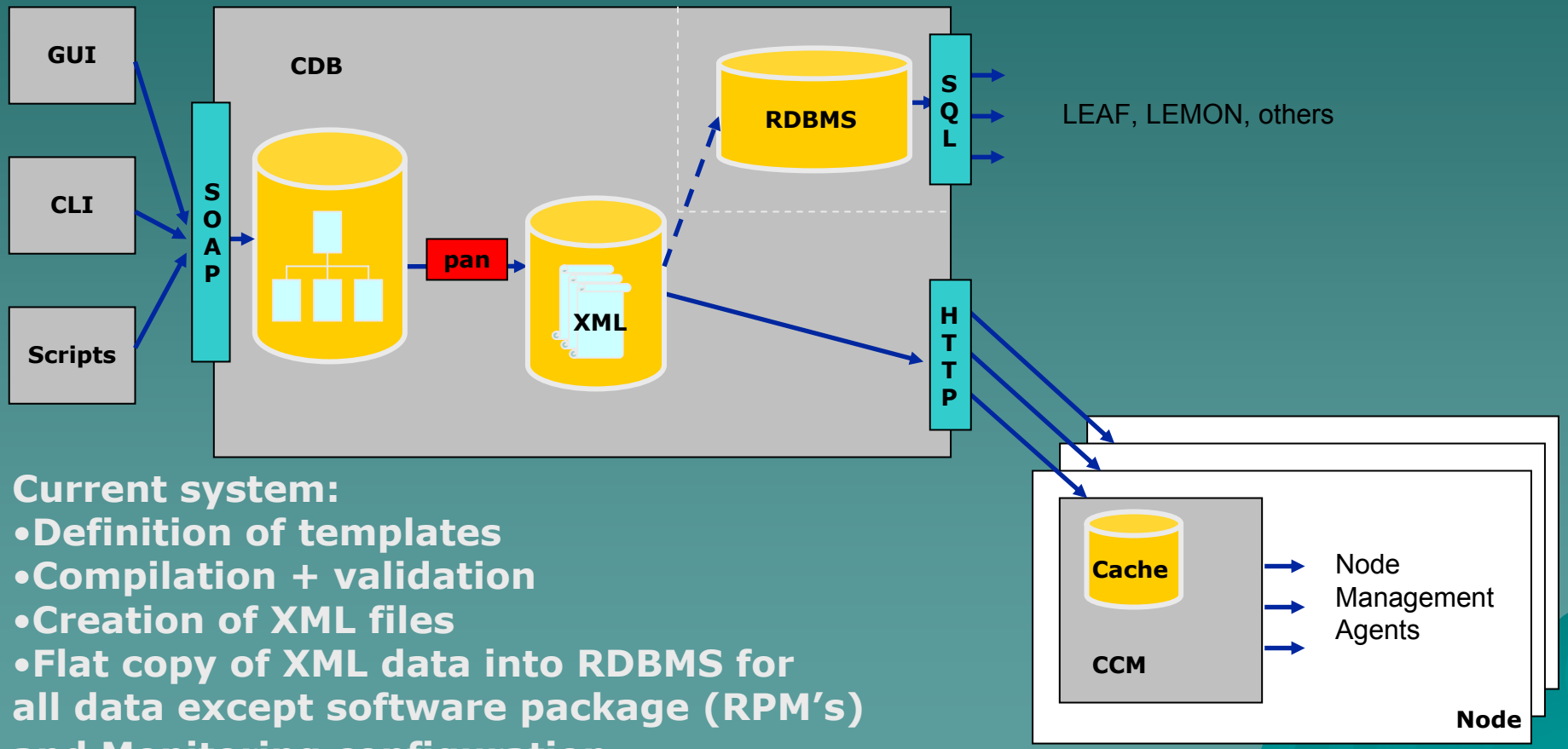
A. Automated updating of configuration information
B. Data privacy
C. Inventory

An RDBMS to replace PAN ?

- Current system: PAN
  (Source: http://hep-proj-grid-fabric-config.web.cern.ch/hep-proj-grid-fabric-config/slides/lisa-06112002/ Lionel Cons)
  - PAN = "high-level description language to describe system configurations"
  - Why a new language? To fulfill requirements and/or preferences:
    - High-level description
    - Avoid information duplication
    - Declarative specification
    - Distributed administration
    - Powerful validation
    - Domain neutral

Configuration Database

(Slide from http://gcancio.home.cern.ch/gcancio/grid/taipei/taipei-elfms.ppt by German)



Current system:

- **Definition of templates**
- **Compilation + validation**
- **Creation of XML files**
- **Flat copy of XML data into RDBMS for all data except software package (RPM's) and Monitoring configuration**

PAN Templates

- Template examples (see following slides)
  - To illustrate the features of PAN
  - For persons not familiar with PAN
  - In particular, to see how configuration data can be organised in hierarchy, with inheritance and specialisation (overwriting)
  - How users can define new variables, functions, data types,…
  - How data can be validated before “commit”

PAN Templates (1)

- **Example: configuration of node "tbed007d"**

```
object template profile_tbed007d;

  include pro_declaration_profile_base;

  include pro_hardware_fileserver_elonex_800_ez;
  include pro_type_fileserver_generic7;
  include netinfo_tbed007d;
  include diskinfo_tbed007d;

  "/hardware/serialnumber" = "CH435-109-13";
  "/hardware/cards/nic/0/hwid"  = "00-02-E3-00-3B-16";
```

- Grouping of data, host-independent, re-used by all similar hosts
- Adding of host-dependent data

PAN Templates (2)

```
template pro_hardware_fileserver_elonex_800_ez;
   "/hardware/model" = "ez";
   [...]
   "/hardware/disks/_3ware_escalade/_0/_0/serialnumber" = "";
   "/hardware/disks/_3ware_escalade/_0/_0/model"
      ="QUANTUM FIREBALLP AS20.5";
   "/hardware/disks/_3ware_escalade/_0/_0/capacity"=20.54*GB;
   "/hardware/disks/_3ware_escalade/_0/_0/manufacturer"
      ="QUANTUM";
   [...]
   "/hardware/disks/_3ware_escalade/_2/_7/serialnumber" = "";
   [...]
   "/hardware/disks/_3ware_escalade/_2/_7/manufacturer"="IBM";
```

- Pre-defined fields, overwriten later (i.e. down)
- User-defined variables

PAN Templates (3)

declaration template **pro_declaration_functions**;

include pro_declaration_functions_general;
include pro_declaration_acl_function;

define variable MB = 1;
define variable GB = 1024 * MB;
define variable TB = 1024 * GB;

- User-defined variables, functions

PAN Templates (4)

template **diskinfo_tbed007d**;

"/hardware/disks/_3ware_escalade/_0/_0/serialnumber" =
"GW0GWF85281";
[…]

- Overwriting

PAN Templates (5)

```
template pro_type_fileserver_generic7;

  [...]
   "/system/cluster/tplname"="pro_type_fileserver_generic7";
   "/software/packages" =
   value("//pro_software_fileserver_generic7/software/packages")
   "/software/packages"= pkg_del("LSF");
   "/software/packages"= pkg_del("CERN-CC-LSF");
   "/software/packages"= pkg_del("lemon-sensor-lsf");
   "/software/packages"={
    if ( value("/hardware/model")=="e0" || [...] ){
         pkg_del("ipmitool");
         pkg_del("ncm-ipmi");
  } else { value("/software/packages");};};
```

- If/then control structures

PAN Template (6)

```
object template pro_software_fileserver_generic7;

include pro_declaration_functions;

include pro_software_packages_cern_redhat7_3_release;
include pro_software_packages_cern_redhat7_3_asis_base;
include pro_software_packages_cern_redhat7_3_cerncc_base;
include pro_software_packages_cern_redhat7_3_quattor;

"/software/packages"=pkg_del("CASTOR-client");
"/software/packages"=pkg_add("CASTOR-disk_server","1.7.1.5-1","i386");
[…]
"/software/packages"=resolve_pkg_rep(value("/software/repositories"));
"/software/repositories"=purge_rep_list(value("/software/packages"));
```

- Grouping, adding

PAN Templates (7)

```
template pro_software_packages_cern_redhat7_3_release;

    include pro_os_redhat7_3;

    "/software/packages"=pkg_add("4Suite");
    […]
    "/software/packages"=pkg_add("XFree86");
    "/software/packages"=pkg_add("XFree86-100dpi-fonts",
    "4.2.1-13.73.23","i386");
```

```
declaration template pro_declaration_functions_general;
    […]
    define function pkg_add = {
    […]
    if (exists(package_default[u_name][0])) {…}
    else { error("no default version for package:"+name);};
   […]
```

- Validation functions

What we like PAN for:

- PAN allows to define data schema in an extremely flexible way, allows fast developments
- Many developers can work at the same time on different templates, with minimal interference
- Templates are human readable, relatively easy to understand and to modify both at the data schema level and at the data values level (difficult though to find the way through the 'include's if not really familiar with the templates)

PAN fulfillment of the Requirements :

- Requirements 1 to 22 are satisfied (14-history partly), **but**
  - (4) Inheritance: ('include' statements) Hierarchy traversal is implicit, user does not need to explicitly express *how* to traverse the hierarchy, but inverting the order of the 'include' may have unexpected consequences
  - (8) Schema: fields can easily be abused, mis-used (related to the fact that there is no "database super user" or coordinator)
  - (8) Schema: fields mandatory, but values set to "---" or "?" or "undefined" …

PAN fulfillment of the Requirements (continued):

- (18) data read access: the CDBSQL schema is a flat copy of (part of) the XML (low level) information
  - Loss of the hierarchical information
    - See for example artificial field "/system/cluster/tplname"="pro_type_fileserver_generic7" in pro_type templates
  - Loss of information for clusters with no host (i.e. no XML file)
    - Cannot use CDBSQL as input for tools where a cluster selection is required, for ex.
  - Schema not normalised:
    - duplication of information
    - Requires implementation and maintenance of VIEWS

PAN fulfillment of the Requirements (continued):

- (18) data read access:
  - the CDBSQL does not contain all the information stored in the XML files (RPMs and Monitoring information is missing because it represents too much info, because not normalised)

- Requirements A to C are not satisfied:
  A. Automated updating of configuration information
  B. Data privacy
  C. Inventory

PAN is nice, But:

- See previous slide for requirements not satisfied
- In particular point A: maintenance of templates is very tedious, very difficult to automate
- Also point C: Inventory data, where to store it ? If in a separate DB, how to insure data consistency ?
- High PAN flexibility can lead to quite a disorder in how the templates are organised and implemented (working on 'trust relationship' may be nice, but strict constraints and rules might be profitable on a longer term basis)
- How to link CDB data with data contained in other DB's (LANDB, …) and keep consistency ?
- PAN is a language invented (~3 years ago) and used for template definition only (non-standard):
  - Support, maintenance issues
- Most of the applications use CDBSQL as source of the data
  - CDBSQL update from the XML files takes an amount of time > 0
  - CDBSQL is read-only: can not be used for fast updates
  - Note: We have already moved the "state" from the templates into RDBMS for that purpose
  - CDBSQL misses some information (see slide 17)

Moving to an RDBMS (ORACLE)

- Would solve the problems listed previously, in particular:
  - Full data access
  - Data easy access for both read and update
  - Link with other DB's (and data consistency)
  - Data privacy
- Would allow to filter the data that goes to the XML files (not all information is needed by the clients)
- Would profit from built-in XML functionality
- Could profit from good ORACLE support at CERN, for optimisation issues, performance tuning

ORACLE DB fulfillment of the Requirements :

1. Content scalability **yes**
   If the schema is correctly designed, CDB data does not represent so much data
2. Grouping of data (for re-use) **yes** (see SW example)
3. Hierarchy **yes** (see SW ex.)
4. Inheritance (for no duplication of data) **yes** (see SW ex.)
5. Overwriting **yes** (see SW ex.)
6. Data types (basic, compound, user-defined) **yes**
   Built-in types and tables
7. Validation **yes**
   Stored-procedures and triggers
8. Schema:
   1. evolution : restriction on easiness for schema evolution (extension is easier than evolution) : **+/-**
   2. fields optional/obligatory **yes**
9. Data transformation functions **yes**
   with views
10. Extensibility (for schema, data types, functions) **yes**

ORACLE DB fulfillment of the Requirements (continued):

11. Consistency **yes**
    For schema and function/procedure changes, a 'recompilation' of all XML files and comparison with previous version is probably a good validation test. How to automate it ?
12. Transactions **yes**
13. Rollback **yes**
14. History **yes**
    can be stored in the DB
15. CDB user scalability
    - CDB administrator: coordination is necessary
    - CDB user: **yes**
    - Protection at table level (GRANT option)
    - Protection at level of Rows in a table also possible via views (Ask ORACLE experts)
16. Abstraction: **no**
    For schema and procedure modifications, the user need to know SQL and PL/SQL, or ask a DBA to do the work

ORACLE DB fulfillment of the Requirements (continued):

17. Content portability (not a CERN requirement): **no (?)**
18. Data read access **yes**
    (and should be done with views)
19. Adding of new (sub) structures **yes**
    But again, requires (PL/)SQL knowledge. This point is related to point 16
20. Modification performance **yes**
    by construction of the product (ORACLE), tuning possible, expertise available
21. Software availability and portability (not a CERN requirement): **no (?)**
22. User interfaces **yes**

A. Automated updating of configuration information **yes**
B. Data privacy **yes**
C. Inventory **yes**

RDBMS implementation example: Software configuration of hosts

- Why SW configuration ?
  - It is not in CDBSQL
  - It uses grouping, hierarchy, inheritance, overwriting
- SW configuration = definition of the set of RPM's to be installed on each node, knowing that
  - nodes being part of a same cluster get the same set of RPM's except specified otherwise
  - Some RPM's may have to be removed depending on the hardware specifications
  - The version of the RPM's to be used may be either 'hardcoded' or defined from a list of default versions

SW Configuration: an RDBMS Prototype

- Prototype:
  - Not everything is implemented
    - left out the repository information and the NCPU dependency
    - No user-interface implemented, but will give examples of interesting queries
  - Reached essentially correct configuration
    - test case = tbed007d (special case, HW dependency)
    - Understood reasons where correctness not reached (but didn't want to spend more time) (see slide 29)
  - Simple schema
    - 3-level hierarchy:
      1. Set of RPMS
      2. Cluster of Sets
      3. Node specialisation
    - Can be made more complex if required (for ex. For a N-level inheritance)

SW Configuration: Table Definition (1)

- RPM's defined by a name, a version, an architecture, a filename
- RPM name shared by many RPM's
  - ⇨Table ARPM (ID, Name)
- Architecture shared by many RPM's
  - ⇨Table ARCH (ID,Name)
- ⇨Table RPM (ID,Name,Version,ARCHID,ARPMID)

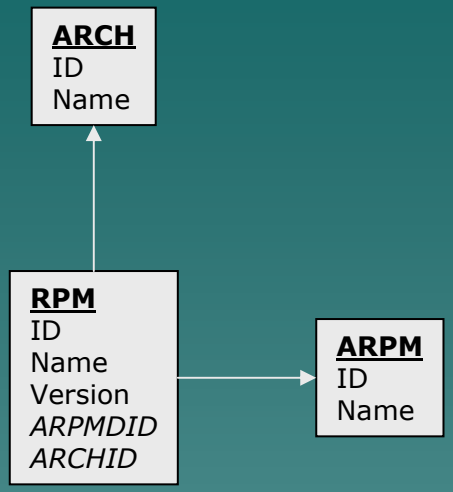

ARCH
ID
Name
RPM
ID
Name
Version
ARPMDID
ARCHID
ARPM
ID
Name

Table content extract: ARPM

ID	NAME
1	ASIS_rpmt
2	4Suite
3	CASTOR-disk_server
4	CASTOR-client
5	CERN-CC-arcd_configuration
6	CASTOR-stager
7	CASTOR-tape_server
8	CERN-CC-3dmd
9	CERN-CC-ACL-CDBServer
10	CERN-CC-ACL-WebServer

Table content extract: ARCH

ID	NAME
1	i386
2	noarch
3	i686
4	i586
5	athlon
6	i486
7	ia64
8	x86_64

Table content extract: RPM

ID	NAME	VERSION	ARPMID	ARCHID
1	ASIS_rpmt-0.3.3-1-i386	0.3.3-1	1	1
2	4Suite-0.11-2-i386	0.11-2	2	1
3	CASTOR-disk_server-1.5.2.1-15-i386	1.5.2.1-15	3	1
4	CASTOR-client-1.5.2.1-15-i386	1.5.2.1-15	4	1
5	CASTOR-client-1.5.2.3-1-i386	1.5.2.3-1	4	1
6	CASTOR-client-1.5.2.5-1-i386	1.5.2.5-1	4	1
7	CERN-CC-arcd_configuration-1.0-1-noarch	1.0-1	5	2
8	CASTOR-disk_server-1.5.2.3-1-i386	1.5.2.3-1	3	1
9	CASTOR-disk_server-1.5.2.5-1-i386	1.5.2.5-1	3	1
10	CASTOR-stager-1.5.2.1-15-i386	1.5.2.1-15	6	1

SW Configuration: Table Definition (2)

- Set of RPM's (= level 1)
  - Like 'pro_software_packages_slc3_*.tpl'
  - Definition of a set of RPM's, for a given OS
    - Difference with templates: here only 'add', no 'replace', no 'delete', otherwise final result depends on order of inclusion of the sets. Only one such case now in CDB, which can be 'cleaned'.
  - Flag each RPM with 'multi' if more than one version is allowed
  - Flag each RPM with 'usedef' if default version to be used
- Table OS(ID,Name)
- Table SetOfRpms (ID,Name,OSID)
- Table SetRpmMap(ID,SetID,RPMID,multi,usedef)

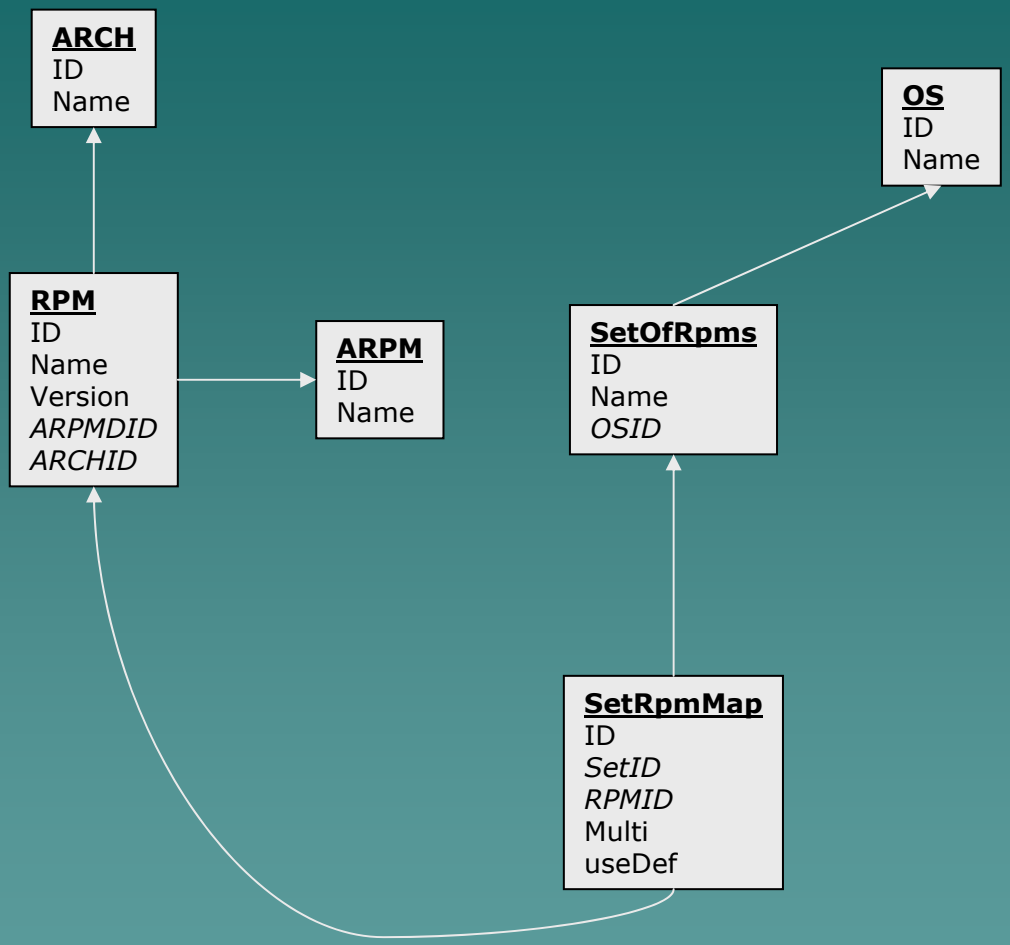

ARCH
ID
Name
OS
ID
Name
RPM
ID
Name
Version
ARPMDID
ARCHID
ARPM
ID
Name
SetOfRpms
ID
Name
OSID
SetRpmMap
ID
SetID
RPMID
Multi
useDef

Table content extract: OS

```
ID  NAME
1   redhat21ES
2   redhat73
3   rhes3
4   slc3
```

Table content extract: SetOfRpms

ID	NAME	OSID
1	pro_software_packages_cern_ia64_slc3_quattor.tpl	4
2	pro_software_packages_cern_ia64_slc3_release.tpl	4
3	pro_software_packages_cern_redhat21ES_quattor.tpl	1
4	pro_software_packages_cern_redhat21ES_release.tpl	1
5	pro_software_packages_cern_redhat7_3_asis_base.tpl	2
6	pro_software_packages_cern_redhat7_3_cerncc_base.tpl	2
7	pro_software_packages_cern_redhat7_3_interactive.tpl	2
8	pro_software_packages_cern_redhat7_3_lcg2_base.tpl	2
9	pro_software_packages_cern_redhat7_3_lcg2_ca.tpl	2
10	pro_software_packages_cern_redhat7_3_lcg2_quattor.tpl	2

Table content extract: SetRpmMap

ID	SETID	RPMID	MULTI	USEDEF
1	1	7631	0	0
2	1	2685	0	0
3	1	7645	0	0
4	1	7612	0	0
5	1	7646	0	0
6	1	7592	0	0
7	1	2492	0	0
8	1	7608	0	0
9	1	5715	0	0
10	1	7620	0	0

SW Configuration: Table Definition (3)

- Default RPM versions are stored in a map, for each OS:
  table
  RPM_OS_DEF (ID, ARPMID, RPMID, OSID)

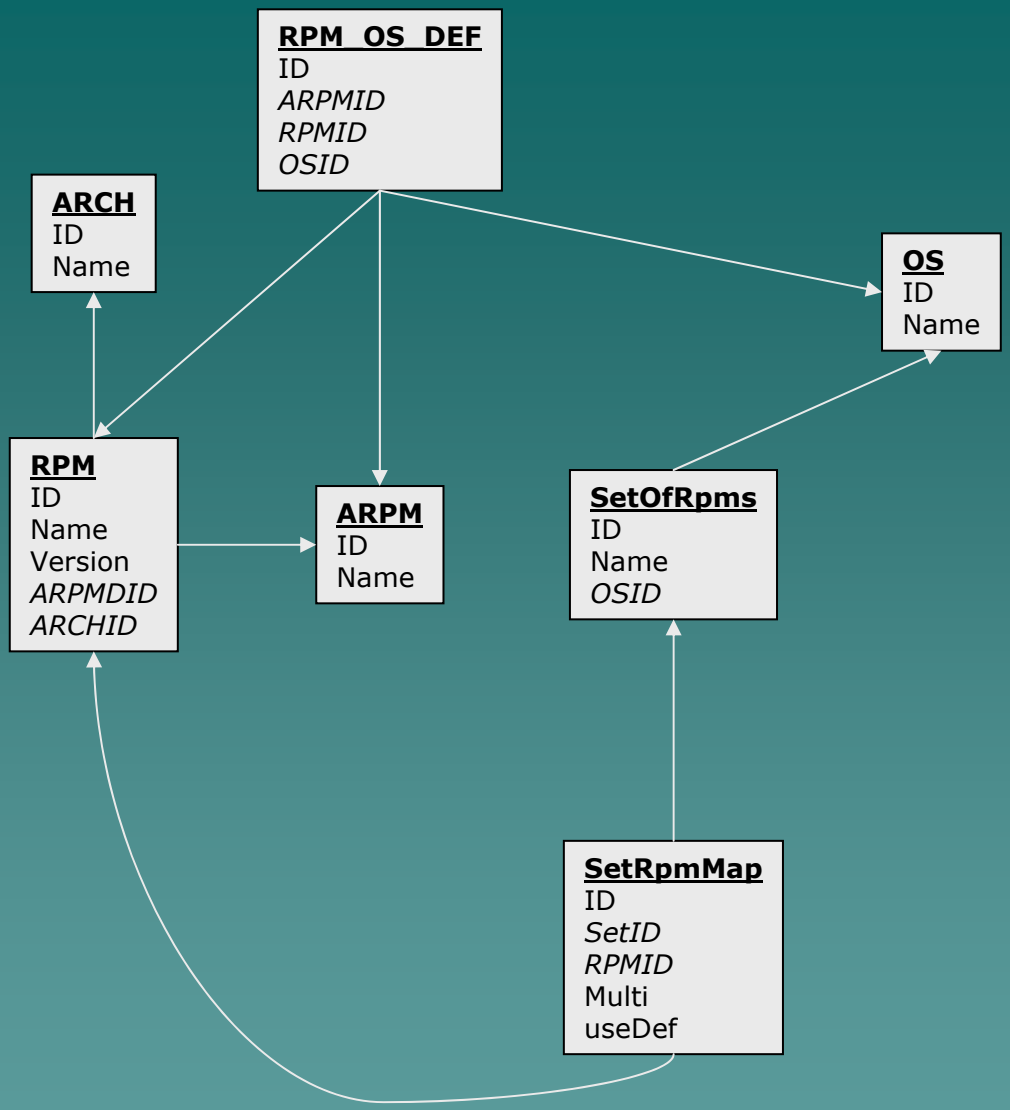

RPM_OS_DEF
ID
ARPMID
RPMID
OSID
ARCH
ID
Name
OS
ID
Name
RPM
ID
Name
Version
ARPMDID
ARCHID
ARPM
ID
Name
SetOfRpms
ID
Name
OSID
SetRpmMap
ID
SetID
RPMID
Multi
useDef

Table content extract: RPM_OS_DEF

ID	ARPMID	RPMID	OSID
1	2	2	1
2	49	154	1
3	50	155	1
4	51	156	1
5	52	157	1
6	53	158	1
7	38	97	1
8	54	159	1
9	55	160	1
10	56	161	1

SW Configuration: Table Definition (4)

- Cluster SW configuration (= level 2)
  - Like 'pro_software_lxbatch_slc3.tpl', plus the SW configuration data possibly found in 'pro_type_*.tpl'
  - Defines the list of RPM's needed for that cluster:
    - Sum of RPM's contained in included Sets ('SetID')
    - Minus possibly some RPM's ('ARPMID')
      - Possibly in function of some HW condition ('HWID','NCPU')
    - Plus possibly some other RPM's ('RPMID','multi','usedef')
    - Replacement of RPM's possible = Minus followed by Plus
- Table ClusterOfRpms (ID,Name,OSID)
- Table ClusterRpmMap (ID, ClusterOfRpmsID, SetID, ARPMID, RPMID, multi, usedef, HWID, NCPU)

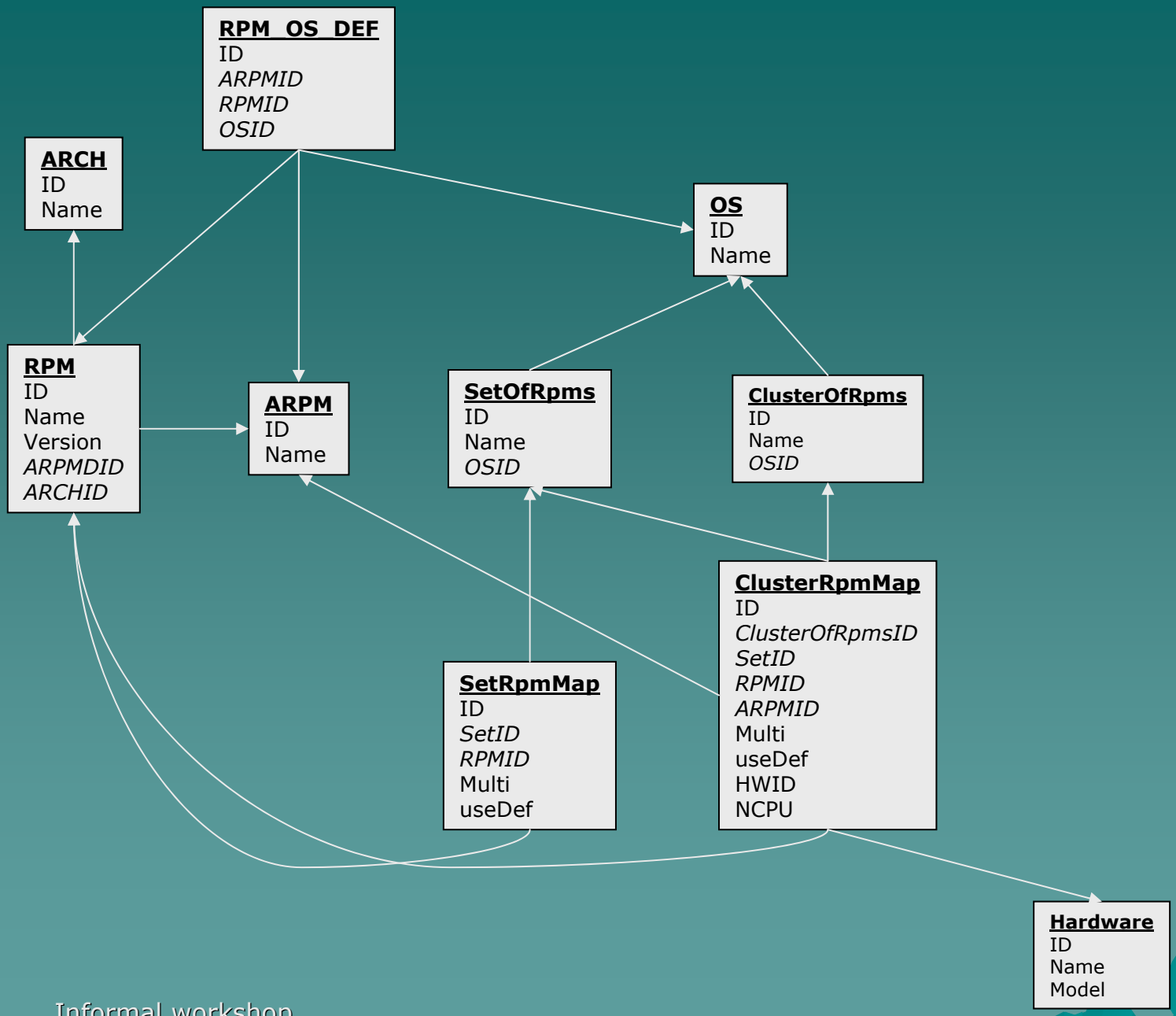

RPM_OS_DEF
ID
ARPMID
RPMID
OSID
ARCH
ID
Name
OS
ID
Name
RPM
ID
Name
Version
ARPMDID
ARCHID
ARPM
ID
Name
SetOfRpms
ID
Name
OSID
ClusterOfRpms
ID
Name
OSID
ClusterRpmMap
ID
ClusterOfRpmsID
SetID
RPMID
ARPMID
Multi
useDef
HWID
NCPU
SetRpmMap
ID
SetID
RPMID
Multi
useDef
Hardware
ID
Name
Model

Table content extract: ClusterOfRpms

ID	NAME	OSID
1	pro_software_atljpgrd7.tpl	2
2	pro_software_castoradm7.tpl	2
3	pro_software_castorgrid7.tpl	2
4	pro_software_castorgrid_slc3.tpl	4
5	pro_software_castorsrv7.tpl	2
6	pro_software_castorsrvES.tpl	1
7	pro_software_castorsrv_slc3.tpl	4
8	pro_software_cvs7.tpl	2
9	pro_software_dbserverES.tpl	1
10	pro_software_dbserver_rhes3.tpl	3
11	pro_software_dbserver_slc3.tpl	4
12	pro_software_fileserver_generic7.tpl	2
13	pro_software_fileserver_generic_slc3.tpl	4

Table content extract: ClusterRpmMap

ID	CLUSTERID	SETID	RPMID	ARPMID	HWID	USEDEF	MULTI	NCPU
73	12	12						
74	12	5						
75	12	6						
76	12	11						
77	12		2927			0	0	
78	12		5954			0	0	
79	12		2705			0	0	
80	12		6260			0	0	
81	12		6315			0	0	
82	12		5701			0	0	
83	12		5905			0	0	
84	12		2322			0	0	
85	12		2353			0	0	
86	12		5953			0	0	
87	12		5906			0	0	
88	12		2225			0	0	
89	12			4				
90	12			1229				
689	12			2285	24			
690	12			2285	27			
691	12			2285	28			
692	12			2285	29			
693	12			2285	32			
694	12			2284	24			
695	12			2284	27			
696	12			2284	28			
697	12			2284	29			
698	12			2284	32			
699	12			2283	24			
700	12			2283	27			
701	12			2283	28			
702	12			2283	29			
703	12			2283	32			
704	12			68				
705	12			12				
706	12			1294				

SW Configuration: Table Definition (5)

- Node SW configuration (= level 3)
  - SW configuration found in 'profile_<node>.tpl'
  - Defines the list of RPM's needed for that node:
    - Sum of RPM's contained in its Cluster ('ClusterOfRpmsID')
    - Modified in the same way as for Clusters (Delete, add, replace=delete+add)
- Table NodeRpms (ID,Name,OSID)
- Table NodeRpmMap (ID, NodeRpmsID, ClusterOfRpmsID, ARPMID, RPMID, multi, usedef, HWID, NCPU)

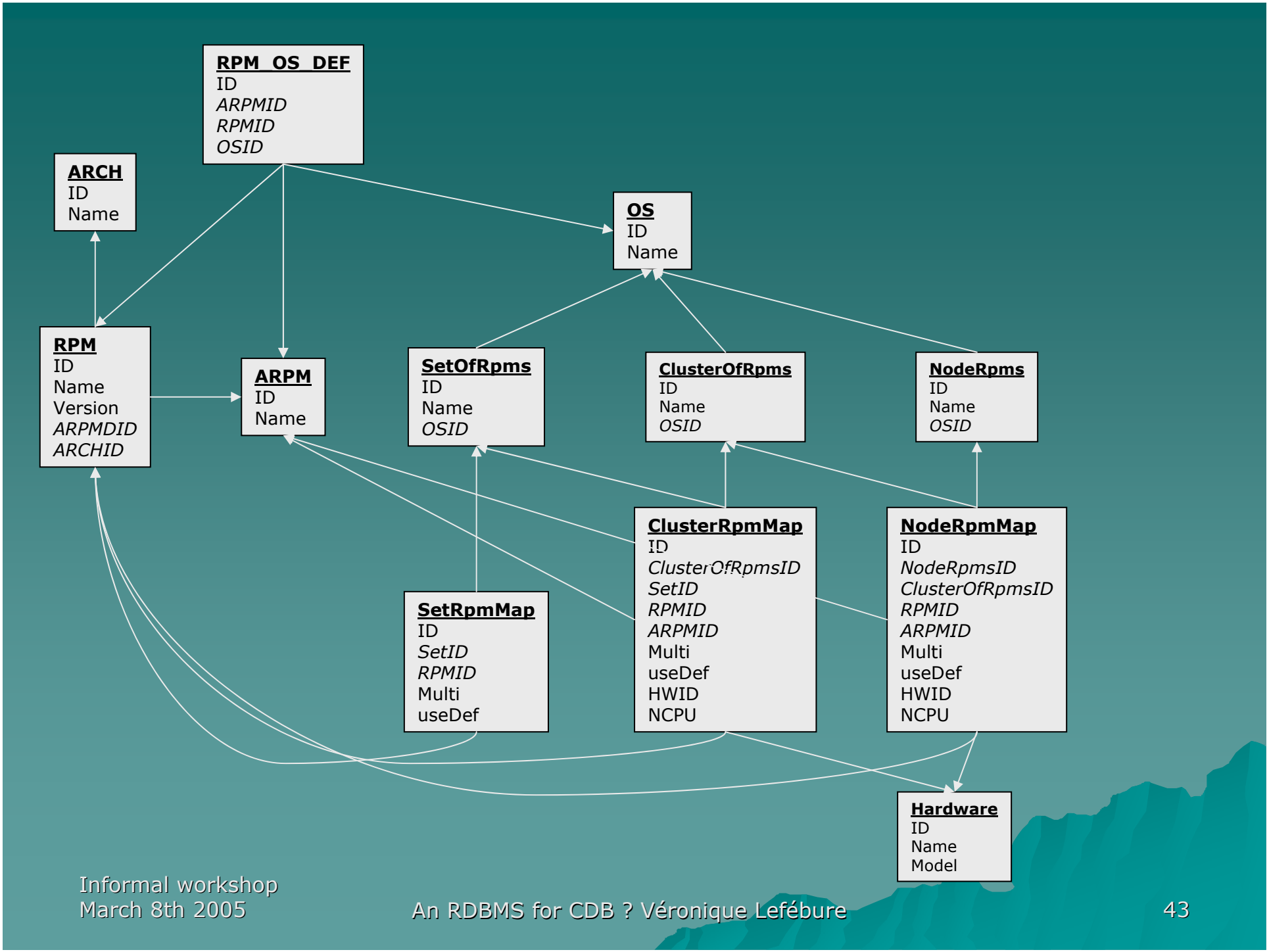

RPM_OS_DEF
ID
ARPMID
RPMID
OSID
ARCH
ID
Name
OS
ID
Name
RPM
ID
Name
Version
ARPMDID
ARCHID
ARPM
ID
Name
SetOfRpms
ID
Name
OSID
ClusterOfRpms
ID
Name
OSID
NodeRpms
ID
Name
OSID
ClusterRpmMap
ID
ClusterOfRpmsID
SetID
RPMID
ARPMID
Multi
useDef
HWID
NCPU
NodeRpmMap
ID
NodeRpmsID
ClusterOfRpmsID
RPMID
ARPMID
Multi
useDef
HWID
NCPU
SetRpmMap
ID
SetID
RPMID
Multi
useDef
Hardware
ID
Name
Model

Table content extract: NodeRpms

ID	NAME	OSID
7	profile_lxb0007.tpl	
64	profile_lxb0070.tpl	
65	profile_lxb0071.tpl	
66	profile_lxb0072.tpl	
67	profile_lxb0073.tpl	
68	profile_lxb0074.tpl	
69	profile_lxb0075.tpl	
70	profile_lxb0076.tpl	
71	profile_lxb0077.tpl	
72	profile_lxb0078.tpl	
73	profile_lxb0079.tpl	
502	profile_ lxb0614.tpl	
827	profile_tbed0007.tpl	
894	profile_tbed007d.tpl	

(OSID not defined because not used yet: default RPM versions not yet used at the node level in CDB)

Table content extract: NodeRpmMap

ID	NODERPMSID	RPMID	ARPMID	CLUSTERID	MULTI	HWID	NCPU	USEDEF
502	502			22				
503	502	2284			1			0
504	502	2282			0			0
505	502	2281			0			0
506	502	2280			0			0
507	502	2277			1			0
508	502	2279			0			0
509	502	2278			1			0
510	502		861					
511	502		864					
512	502		862					
513	502		863					
514	502		604					
515	502		597					
516	502		570					
947	894			12				

SW Configuration: Table Definition (6)

- Hardware:
  - simplified for this exercise
  - NCPU ignored

Table Hardware (ID, Name, Model)

Table content extract: Hardware

ID	NAME	MODEL
1	pro_hardware_cvs_elonex_600.tpl	Elonex_600MHz
2	pro_hardware_cvs_seil_2003_1.tpl	SEIL_2.4GHz
3	pro_hardware_diskarray_transtec_6100_t0.tpl	t0
4	pro_hardware_elonex_2800.tpl	Elonex_2.8GHz
5	pro_hardware_elonex_450_PII.tpl	Elonex_550MHz
6	pro_hardware_elonex_450.tpl	Elonex_550MHz
7	pro_hardware_elonex_500.tpl	Elonex_500MHz
13	pro_hardware_elonex_single_2660.tpl	Elonex_2.66GHz
14	pro_hardware_fileserver_cogestra_500_c0.tpl	c0
15	pro_hardware_fileserver_cogestra_600_c1.tpl	c1
16	pro_hardware_fileserver_elonex_1100_e1.tpl	e1
17	pro_hardware_fileserver_elonex_2000_e2.tpl	e2
34	pro_hardware_lxeng_elonex_2600.tpl	2600
35	pro_hardware_lxserv_seil_2400.tpl	SEIL_2.4GHz
52	pro_hardware_tapeserver_fake2_800.tpl	misc

SW Configuration: Table Definition (7)

- Node:
  - simplified for this exercise

Table Node (ID, Name, HWID, TypeID, NodeRpmsID)

- Type:
  - Also simplified here
  - Should contain references to configuration for other domains such as System, Components, Monitoring, OS

Table Type (ID, Name, ClusterOfRpmsID)

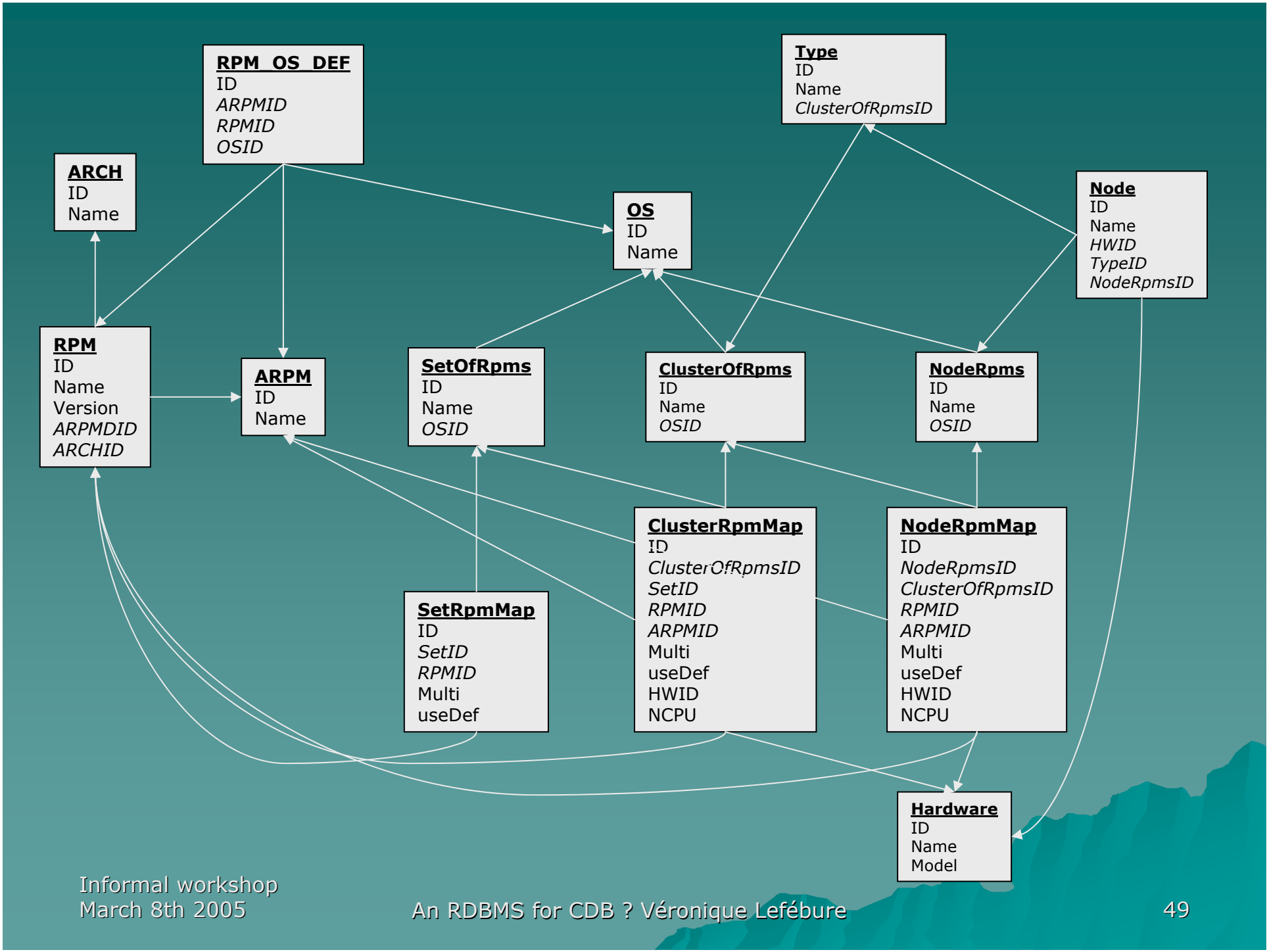

RPM_OS_DEF
ID
ARPMID
RPMID
OSID
Type
ID
Name
ClusterOfRpmsID
ARCH
ID
Name
OS
ID
Name
Node
ID
Name
HWID
TypeID
NodeRpmsID
RPM
ID
Name
Version
ARPMDID
ARCHID
ARPM
ID
Name
SetOfRpms
ID
Name
OSID
ClusterOfRpms
ID
Name
OSID
NodeRpms
ID
Name
OSID
ClusterRpmMap
ID
ClusterOfRpmsID
SetID
RPMID
ARPMID
Multi
useDef
HWID
NCPU
NodeRpmMap
ID
NodeRpmsID
ClusterOfRpmsID
RPMID
ARPMID
Multi
useDef
HWID
NCPU
SetRpmMap
ID
SetID
RPMID
Multi
useDef
Hardware
ID
Name
Model

Table content extract: Node

ID	NAME	TYPEID	HWID	NODERPMSID
500	profile_lxb0612.tpl	2	37	500
501	profile_lxb0613.tpl	2	37	501
502	profile_lxb0614.tpl	2	37	502
503	profile_lxb0615.tpl	2	37	503
504	profile_lxb0616.tpl	2	37	504
505	profile_lxb0617.tpl	2	37	505
893	profile_tbed0079.tpl	1	39	893
894	profile_tbed007d.tpl	11	27	894
895	profile_tbed0080.tpl	1	39	895
896	profile_tbed0081.tpl	1	39	896
897	profile_tbed0082.tpl	1	39	897
898	profile_tbed0083.tpl	1	39	898
899	profile_tbed0084.tpl	1	39	899
900	profile_tbed008d.tpl	11	27	900

Table content extract: Type

ID	NAME	CLUSTEROFRPMSID
1	pro_type_lxbatch_slc3.tpl	24
2	pro_type_lxbatch7.tpl	22
3	pro_type_lxdb_slc3.tpl	30
4	pro_type_lxdb7.tpl	29
5	pro_type_lxnoq.tpl	22
6	pro_type_dbserverES.tpl	9
7	pro_type_lxjra1dm_slc3.tpl	39
8	pro_type_lxjra1prot_slc3.tpl	40
9	pro_type_lxjra1test_slc3.tpl	41
10	pro_type_lxbatch7LCG2.tpl	21
11	pro_type_fileserver_generic7.tpl	12
12	pro_type_lxshare7.tpl	48

List of RPMs Resolved

- View ALLRPMS (RPMID, NodeRpmsID)
- Hides the query:

```
(SELECT SetRpmMap.rpmid,NodeRpmMap.NodeRpmsID FROM SetRpmMap, NodeRpmMap,
ClusterRpmMap WHERE SetRpmMap.SetID = ClusterRpmMap.SetID AND
ClusterRpmMap.ClusterOfRpmsID =NodeRpmMap.ClusterOfRpmsID AND
NodeRpmMap.ClusterOfRpmsID is Not Null)
MINUS (SELECT RPM.id, NodeRpmMap.NodeRpmsID FROM ClusterRpmMap, RPM, NodeRpmMap
WHERE ClusterRpmMap.ClusterOfRpmsID = NodeRpmMap.ClusterOfRpmsID AND
ClusterRpmMap.ARPMID =RPM.ARPMID AND ClusterRpmMap.ARPMID is not null AND
ClusterRpmMap.HWID is null )
UNION ALL (SELECT ClusterRpmMap.rpmid, NodeRpmMap.NodeRpmsID FROM ClusterRpmMap,
NodeRpmMap WHERE ClusterRpmMap.ClusterOfRpmsID = NodeRpmMap.ClusterOfRpmsID AND
ClusterRpmMap.rpmid is not null )
MINUS (SELECT RPM.id,NodeRpmMap.NodeRpmsID FROM ClusterRpmMap, RPM,
NodeRpmMap,Node WHERE ClusterRpmMap.ClusterOfRpmsID = NodeRpmMap.ClusterOfRpmsID
AND ClusterRpmMap.ARPMID = RPM.ARPMID AND ClusterRpmMap.ARPMID is not null AND
ClusterRpmMap.HWID =Node.HWID AND Node.NodeRPmsID= NodeRpmMap.noderpmsID )
MINUS (SELECT RPM.id,NodeRpmMap.NodeRpmsID FROM RPM, NodeRpmMap WHERE
NodeRpmMap.ARPMID=RPM.ARPMID AND NodeRpmMap.ARPMID is not null )
UNION ALL (SELECT rpmid,NodeRpmMap.NodeRpmsID FROM NodeRpmMap WHERE rpmid is not
null )
```

- Currently takes ~0.1 sec per host (query run for 900 hosts)
- DB size: 5MB-900 hosts => 6.5MB-10000 hosts

SW Configuration use-cases (1)

- Trivial use-cases:
  - Insert a new RPM
  - Add/remove/replace an RPM in a Set/Cluster/Node
  - Change default version and propagate to Set/Cluster/Node
  - Create new Set/Cluster of RPMs
  - Move a node from Type1 to Type2

SW Configuration use-cases (2)

- More interesting use-cases:
  1. What is the list of hosts using RPM X version Y ?
  2. Where is RPM X included ?
  3. Where is there an RPM used with a version different from the default one ?
  4. Is the use of multiple versions of the same RPM allowed ? (use of 'multi' flag, in fact more a validation procedure than a use-case)

SW Configuration use-cases:

What is the list of hosts using RPM X version Y ?

```
select Version, Host
  from allrpms_more
 where ARPM='MySQL-client'

VERSION   HOST
4.0.23-0  profile_lxb0771.tpl

Takes ~0.2 sec
```

SW Configuration use-cases:

Where is RPM X included ?

```
select Place, Version
   from rpm_inclusion
 where name='MySQL-client'
```

PLACE	VERSION
pro_software_packages_cern_slc3_lcg2_ce.tpl	4.0.20-sl3
profile_lxb0771.tpl	4.0.23-0

Takes ~0.2 sec

SW Configuration use-cases:

Where is there an RPM used with a version different from the default one ?

```
select RPMName, SetVersion, DefVersion
  from defnotused
 where setname=
  'pro_software_packages_cern_slc3_release_base.tpl'
```

```
RPMNAME      SETVERSION  DEFVERSION
popt         1.8.1-4.4   1.8.2-13
rpm-build    4.2.1-4.4   4.2.3-13
rpm          4.2.1-4.4   4.2.3-13
rpm-devel    4.2.1-4.4   4.2.3-13
rpm-python   4.2.1-4.4   4.2.3-13
```

Takes ~4 sec

SW Configuration use-cases:

Is the use of multiple versions of the same RPM allowed ?

```
select arpm,multi,version from allrpms_more where host = 'profile_lxb0010.tpl'
and arpm in( select arpm from allrpms_more where host = 'profile_lxb0010.tpl' group by
      arpm having count(*)>1    )
```

ARPM	MULTI	VERSION
ant	0	1.5.2-23
ant	0	1.6.1-sl3
edg-utils-system	0	1.6.1-1
edg-utils-system	0	1.6.1-1_sl3
kernel	1	2.4.21-20.EL.cern
kernel	1	2.4.21-27.0.2.EL.cern
kernel	1	2.4.21-20.0.1.EL.cern
kernel-smp	1	2.4.21-20.EL.cern
kernel-smp	1	2.4.21-27.0.2.EL.cern
kernel-smp	1	2.4.21-20.0.1.EL.cern
perl-TermReadKey	0	2.21-1
perl-TermReadKey	0	2.20-12
swig	0	1.1p5-22
swig	0	1.3.19-6.1_sl3

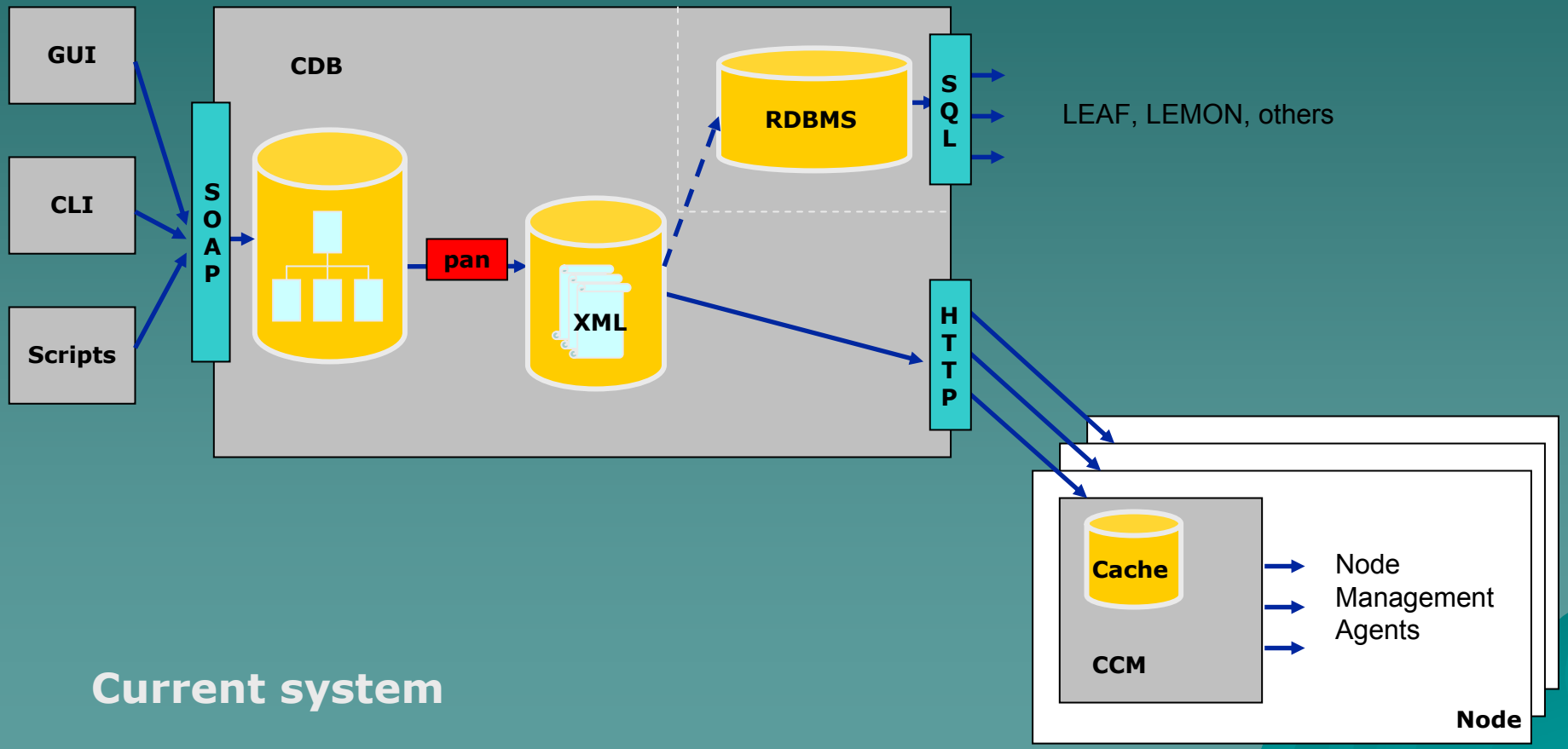
Takes ~0.5 sec

Moving to an RDBMS

- Would cost in manpower (see further)
  - But we already spend a non-negligible amount of manpower for:
    - Implementation, maintenance, testing of tools that parse templates, for automation of updates
    - Support of CDBSQL and creation & maintenance of views
    - Individual template maintenance (‘sed’,’grep’)
- Would cost in loss of some flexibility
  - But we have spent more than 2 years defining the current schema, we should soon reach stability (extension of existing schema is not an issue)
- Would cost in portability within Quattor
  - But is it a high priority requirement ?

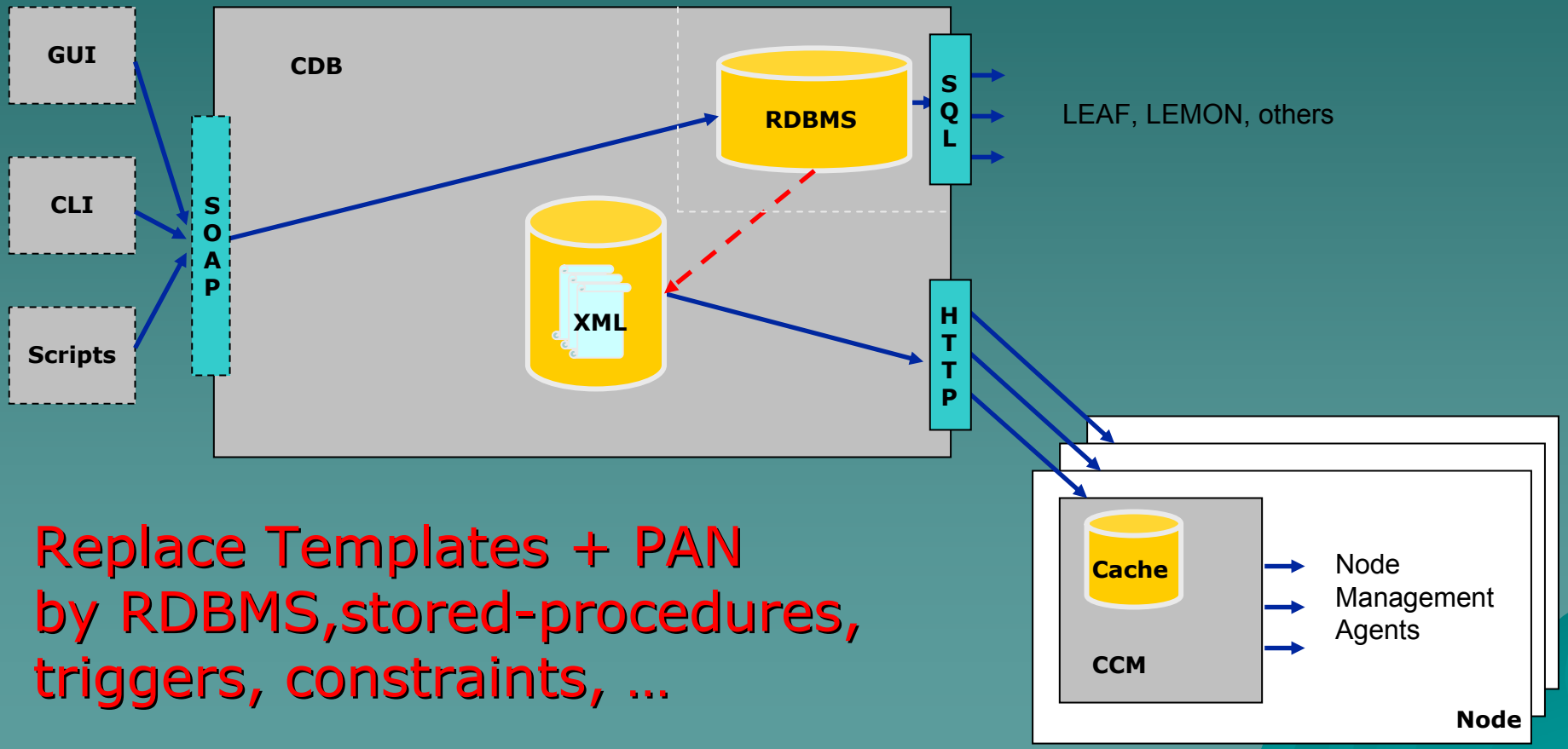
Configuration Database

(Slide from http://gcancio.home.cern.ch/gcancio/grid/taipei/taipei-elfms.ppt by German)



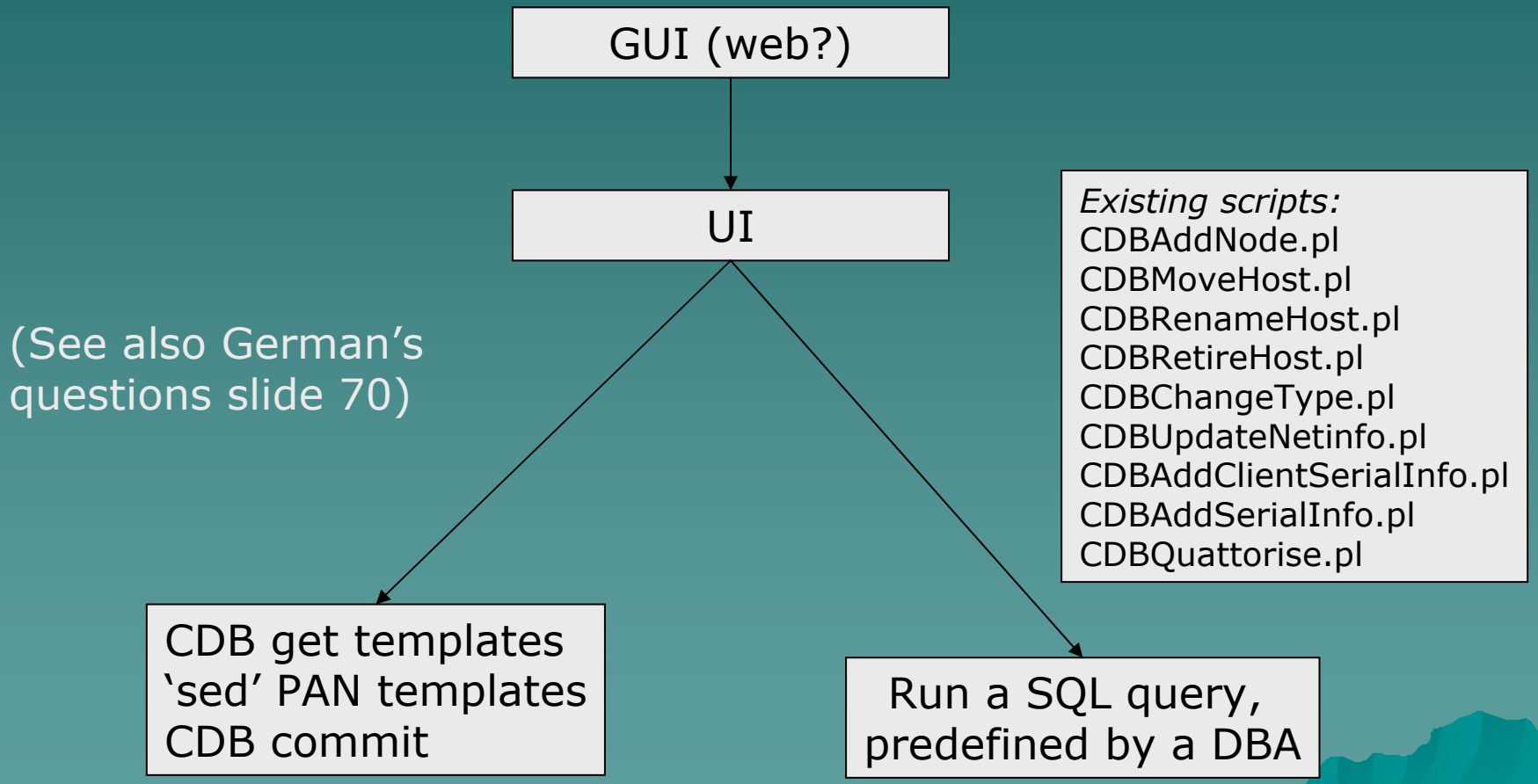
Current system

What exactly would be replaced ?



Replace Templates + PAN
by RDBMS,stored-procedures,
triggers, constraints, …

User-interface for Data Update and Read access



(See also German's questions slide 70)

Estimation of manpower and time:

1. Tasks
2. Tasks split between Domains
3. How to execute each Task
4. Skills needed per Task
5. Amount of time needed per Task

Estimation of manpower and time: Tasks

1. Move data from PAN templates to RDBMS:
    1. Schema definition
    2. Data translation
2. Database content validation
3. Database schema, views, procedures tuning for performance
4. Implementation of a complete user-interface
    1. For data update
    2. For data read-only queries (for information & verification)
5. XML file creation
    1. Mechanism
    2. performance

Estimation of manpower and time: Domains

- Tasks 1 (schema definition) and 4 (user-interface) can be split between the different domains of configuration:
  1. Hardware (and Inventory)
  2. Software (RPM's) + OS/kernel
  3. Monitoring
  4. System
  5. Components
  6. Any new domain that would come
- Interface between the domains can be provided by the mean of VIEWS
- Tasks need to be coordinated and supervised by an "architect" who has a global view

Estimation of manpower and time: How to execute each Task

1. Move data from PAN templates to RDBMS:
    1. Analysis of the current data model (use of documentation if any, else, provide one and submit it for review/validation by the experts)
    2. Requires parsing of the templates, at least in order to reproduce the data grouping (that information is lost in CDBSQL)
2. Database content validation
    1. For intermediate steps of the development:
        1. For SW: compare results with "rpm –qa" output for each machine (SW configuration is not available in CDBSQL)
        2. For the rest: compare results with CDBSQL content (for Monitoring: ?)
    2. When XML files available: comparison of their content
3. Database schema, views, procedures tuning for performance
    1. Keep the update times and the xml-file creation times below a given threshold, for the complete list of hosts
    2. Simulate order of 5-10 times more machines for scalability being insured
    3. After each tuning action, re-validate data
    4. Provide feedback to person(s) working on Task 1
4. Implementation of a complete user-interface
    1. Should make use only of stored-procedures and views provided by Task 1
    2. Performance to be improved by Task 3 if needed
5. XML file creation

Estimation of manpower and time: Skills needed per Task

1. Move data from PAN templates to RDBMS:
   1. Full understanding of PAN and Configuration data is required
   2. A minimum of RDBMS (ORACLE/SQL, PL/SQL) knowledge is needed
   3. Perl (for parsing)
2. Database content validation
   1. Can be performed by anyone, on the basis of the specifications of experts of Task 1
3. Database schema, views, procedures tuning for performance
   1. ORACLE expertise required
4. Implementation of a complete user-interface (in language X)
   1. Need a good understanding of the use-cases
   2. Need knowledge of language X
5. XML file creation
   1. XML and ORACLE (?) knowledge
   2. Understanding of the requirements related to the content and format of the files

Estimation of manpower and time: Amount of Time needed per Task

1. Move data from PAN templates to RDBMS:
   about 3-4 weeks per Domain = 15-20 weeks
2. Database content validation:
   about 2 weeks
3. Database schema, views, procedures tuning for performance:
   about 3 weeks
4. Implementation of a complete user-interface:
   about 2-3 weeks per Domain = 10-15 weeks
5. XML file creation:
   about 2 weeks

Total: about 32-42 weeks, i.e.
between ~1 year for 1 person
and ~4 months for 3-5 persons

German's questions (1)

1. *It is equally important to work towards an architectural description of the application(s) wrapping around the data model. We should not forget that we plan to replace not only a data model or a database backend, but a complete set of applications and modules for configuration management. Having an architecture design document is a paramount prerequisite to any of the tasks listed in slide 64. This architecture document should provide information on the design of the modules replacing our current CDB/Pan, including module specifications and descriptions, relationships and sequence diagrams. Also the detailed data model description (E/R diagrams, data functions, etc) should be part of it.*
    1. The Quattor architecture design document does not changed neither is the 'global schema': apart from where the data is stored and how it is accessed, nothing else would be changed (except maybe for a few restrictions like the N-level hierarchy simplified to a 3-level one).

German's questions (2)

The document should provide solid answers to questions including: (see following points)

2. *What is the CDB-Applications interface? Via SOAP as suggested in p. 61 - if yes, what is the exact API? Or will we use direct SQL statements? Via views, functions? Which ones?*
   1. See slide 62
   2. Interface: (SOAP) (to be (re-)defined at this meeting ?)
   3. a maximum (if not everything) inside the DB (Views and procedures)
   4. More details to be defined at this meeting ?
3. *What replaces the current cdbop command line interface? What about GUI's? How many GUI's will there be?*
   1. data update: API procidedin 4. above
   2. schema update: available tools for DB administration
   3. as less GUIs as needed

German's questions (3)

4. *What replaces the current PAN templates holding functions? How can functions be edited/modified?*
   1. stored-procedures, constraints
   2. edit of them: DB administration tool (other way ?)

5. *How can the hierarchy be modified? How can clusters/sub-clusters be added for node collections? How can information sets (new HW/SW/config descriptions) be added/modified? Can this be done by the service manager or does he need to escalate this to a DBA?*
   1. all standard use-cases (Data update/new entries): API provided (to be implemented), so usable by any body, depending on priviledges

German's questions (4)

6. *How is history implemented for both the data and data transformation functions? How does Oracle's transaction capabilities map into operations like roll back to yesterday's setup of my cluster/node?*
   1. How do we do that now ? we have a CVS tag for the whole configuration, what if we want to roll back for one cluster only?
   2. What does ORACLE offer ?
   3. We can implement our own history mechanism, storing it in the DB itself (ex: instead of modifying entries, copy and archive old entry, leave them available for re-use)
7. *How is a scalable and fast generation of XML profiles achieved, ensuring compatibility with the Quattor 'global schema' conventions?*
   1. XML file generation: ?
   2. 'global schema': dictionnary needed= VIEW

German's questions (5)

8. *What replaces the current dependency engine which minimizes re-validation and XML regeneration?*
    1. dependency = table of list of nodes touched by a change
9. *How is validation propagated down the hierarchy? (Eg. changing a default partitioning rejected because of some nodes with too small hard disks)*
    1. triggers
10. *Where is the mapping between SQL tables and functions and the 'global schema'? How is this mapping updated whenever tables are added/modified/deleted?*
    1. mapping to be done
    2. maintenance by DBA

German's questions (6)

11. *How are ACL's implemented? Eg. how does Oracle's table/row protection mechanism (slide 21) translate into ACL's for cluster, node, hardware, site settings?*
    -views defined as such

    How can I protect my data from erroneous SQL statements?
    -SQL statements defined by DBA/Experts, not by the user

    How will we authenticate, is DB authentication sufficient?
    -ORACLE expert answer: …

The resulting architecture should be evaluated against our requirements and Use Cases, and should be used for work effort estimations. Failing to provide such an architecture document may lead us into a wild growing collection of difficult to maintain and understand scripts, interfaces, GUI's etc.

German's questions (7)

Some specific comments on the slides.

1. *s.15: Comments on schema. There are validation functions and (user defined) typing mechanisms in Pan. For example, a 'string' type can be enhanced to allow only alphanumerical chars; or (valid) IP/Ethernet addresses; or enumerations of valid strings defined in a constant or in a different data field. (However: how is this particular problem solved in Oracle, maybe in a more elegant way?)*
    1. *It is much easier to by-pass constraints in PAN than in ORACLE because everything is accessible to everybody in PAN (for now at least)*
2. *s.16: The schema is normalised (Quattor "global schema"), and all Quattor modules follow this schema. Note that this schema needs to be respected by the relational solution. (What is not normalised are the views on top of the schema, which are just shortcuts to entries to the schema for convenience.)*
    1. *The CDBSQL schema is not normalised*

German's questions (8)

3. *s.20-22: Oracle may provide some foundations which we can use, but the requirements need to be measured against our specific architecture (see above, eg. on rollbacks and ACL's)*
4. *s.23: Including RPM's as a function of the hardware is just an example, and we may be hit by this much more in the future as seen in Grid setups. We need to foresee a flexible mechanism allowing for conditional lookups.*
    1. *RDBMS means use of indexes (pointers): for each new dependency, we have to add a column in a table, and update the corresponding views. Maybe ORACLE experts have more ideas … ?*
5. *s.23: A N-level inheritance is in fact what we have nowadays (think of subsets of LXBATCH used for Alice, LCG, etc).*
    1. *Yes, but Alice in lxbatch is the only case*
    2. *It is being removed because it is causing more troubles than advantages*
    3. *LCG is just another set of RPMs added to lxbatch ones*

German's questions (9)

6. *s.25-51: How does the validation against the SWRep contents work?*
    1. *In the same way as now*
7. *s.52: Does the query change with the number of hierarchies? Eg. what happens if we have nodes resulting of a variable number of hierarchies like we have now? Do we have to use different queries as a function of the number of hierarchies? How can we know which query to use?*
8. *s.64: See above (architecture document)*
9. *s.64: The XML generation engine performance and compatibility is paramount and needs to be addressed as early as possible in the process.*
10. *s.66: See my previous point. The database content validation should be done extracting the XML profiles' SW configuration and not checking against live node contents.*
    1. *Agreed*

Timeline Proposal(s)

- Depends who is available, with which level of expertise regarding CDB and ORACLE

Acknowledgements

- Thanks to German, Bill, Vlado, Tony O., Tony C., Maciej, Thorsten, Nick, Zheska, Eric, Nilo, Michal for usefull discussions and support.