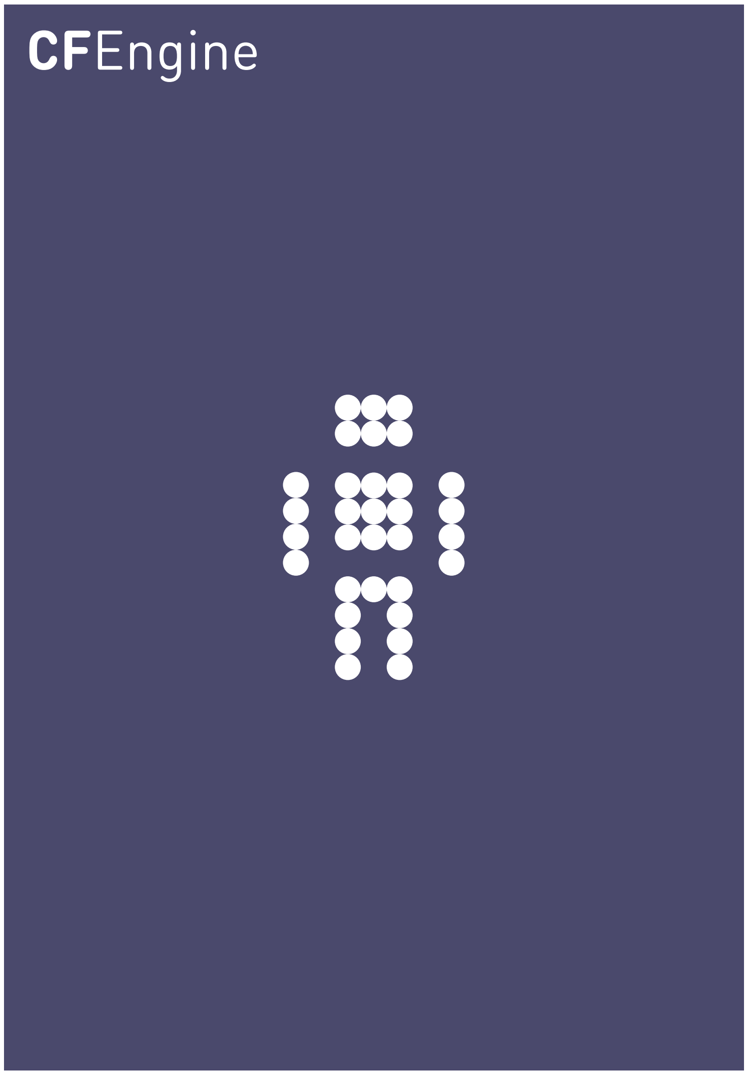


Implementing Knowledge Management

A CFEngine Special Topics Handbook

CFEngine AS

Everyone agrees that Knowledge Management is important, but few really appreciate what it means to manage knowledge, or how to begin. CFEngine's application area is principally operational, but it provides tools for integrating with business development and organizational management.

This short guide suggests how to begin implementing a knowledge management strategy using CFEngine Nova as a centrepiece for integrating data from many different sources.

Table of Contents

What is knowledge management?

Everyone agrees that Knowledge Management is important, but few really appreciate what it means to manage knowledge, or how to begin. Ironically perhaps, even learning institutions (so-called knowledge based industries) like schools and universities often do not do a good job of this. Having a wealth of knowledge is not the same as knowing how to foster and protect it. Most organizations take their knowledge for granted.

In the worst case, poorly run organizations end up as collections of individuals, each of whom possesses knowledge and experience, but none of whom knows anything about the others' roles. This scenario presents a high risk for the continuity of the organization, since taking out even a single player could cripple a key work process.

The end result is that many organizations are caught in a poverty trap of knowledge: they spend their time reacting to emergencies cultivated by a lack of confidence and certainty, and they are too busy patching over holes that they never have time to learn enough to escape the pattern.

The Third Wave of configuration management emphasizes the role of knowledge in coping with the diversity and adaptability that modern organizations crave. This document is a beginning towards thinking about IT management in a new way: as a knowledge resource, or a set of insight-driven intentions that are maintained by smart machinery. This vision is about rehumanizing System Administration through a proper division of labour between Man and Machine.

Risk and uncertainty

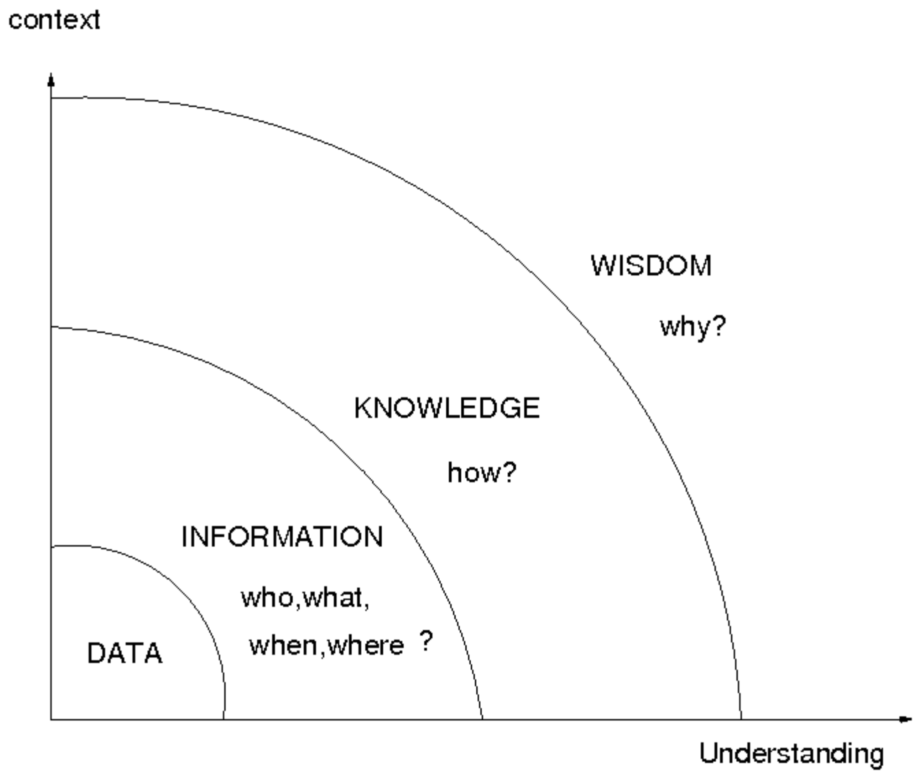
Knowledge exists to reduce our uncertainty about the world. We don't generally learn in order to get smart, or to be experts; rather we learn by necessity and feel smart when there are no remaining surprises. Experts are people who can answer questions they already know the answer to; they are supposedly extra-ordinary. Innovators are people who can solve new problems – they are even rarer.

The main aim of knowledge management is not to make everyone an expert, but to improve the predictability of workflow processes so that we can respond to unpredictable changes in the environment with greater confidence, without losing the feeling of control over the situations we find ourselves in. To do that, we have to make the connections between the appropriate parts to that we can always obtain the answers.

How should you begin?

The road to maturity can be seen as stepping through one or more of the following progressions:

- Data –> Information –> Knowledge –> Wisdom (DIKW)
- Objectives –> Results
- Intentions –> Promises –> Assessments



Each of these sequences represents an increasing level of commitment. You need to have data to interpret them as information. You need to have goals or objectives in order to realize them and see results. You need to have intentions to make promises, and so on. Ask yourself: does everyone in your team know your business goals, and how to translate their efforts into progess?

The CFEngine Knowledge map

Maps guide us to where we need to get to from where we are (even when we don't fully understand where we are). A map places us in the context or a larger picture and helps us to find our way. The Knowledge map is a part of the CFEngine Mission Portal, which in turn is provided with CFEngine Enterprise Edition. Significant instrumentation of the software, user process and policy analysis are provided in these editions in order to make Knowledge mining as transparent to the user as possible.

Our first task in Knowledge Management is to create a map of the knowledge landscape in which we work.

CFEngine contains a tool `cf-know` for creating a map called a Topic Map, that forms to basis of our approach to documentation. Further, CFEngine Enterprise Editions, starting with Nova help fill in the landscape of this map automatically.

To build a map, you will need data (hence the beginning of the DIKW chain). To use a map, you need an objective or an intention, hence the others.

Ask yourself: what are the places in this map, and what are the paths between them? This is a question every organization needs to ask itself.

Pitfalls to avoid

CFEngine does a lot of work for you, but you need to do something to help yourself too. As with all information, garbage in leads to garbage out. Your organization should create documents and references to capture its internal knowledge. If you are doing this well, it should not be a major task.

Knowledge gets out of control when you don't standardize ideas, concepts and procedures. By standardizing topics and concepts, you prevent an explosion of redundancy – only then can you see the signal in the noise. Standardization minimizes the number of things you have to deal with. Be careful though, if you go too far, standardization can become a straightjacket, preventing you from clear and free expression. Too much standardization and you will be stuck in a rut.

Dedicate yourself to a culture of simplicity, but not of over-simplification (which often backfires). Always work to avoid special cases and exceptions, unless they can be rigorously justified.

This is a question of economics: is it worth the cost of maintaining a special case? CFEngine's is designed to make it a lot easier and cheaper to manage exceptions, so you needn't worry too much. Just don't go mad.

Simplification is the hardest thing you will ever do. Anyone can make something more complicated, but it takes real work to simplify something[1]. Einstein's famous quote captures this: *You should always make everything as simple as possible, but no simpler.*

Using the Knowledge Map

CFEngine's exploration of knowledge management is still in its infancy – we are learning constantly about how to use state of the art technologies and techniques. This is still very much an experimental area that we continue to improve upon.

Let's look at the current state of the smart index which we call the (Copernicus) Knowledge Map. If you go to the Mission Portal Library and search for a topic, you will land in the Knowledge Map. You will immediately see a graphical image and number of tabs that contain various different renditions of the information.

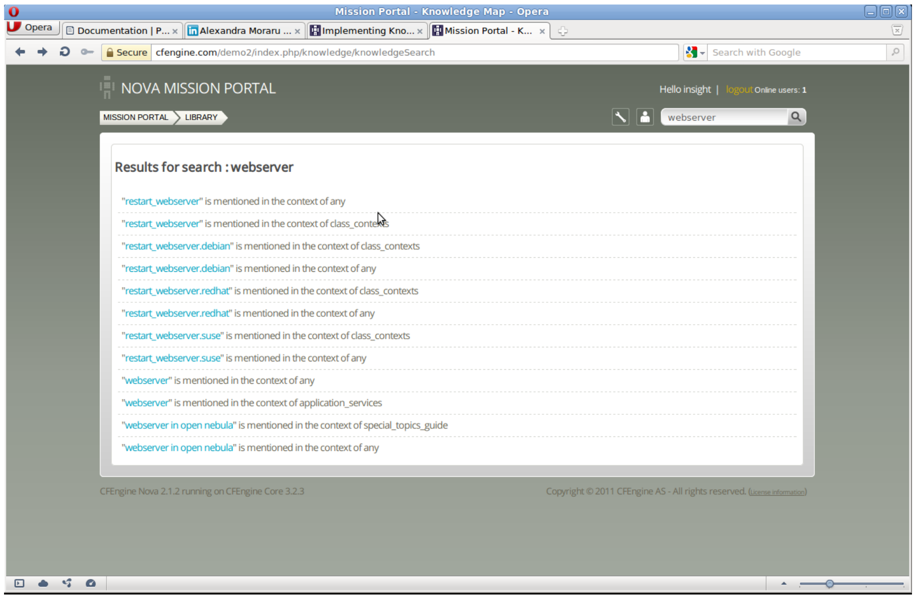
Presently, these are called:

- Map - a visual view of a small neighbourhood of the closest related topics that places the search topic in the context of closely related items. The search topic might not be a the centre of this little universe, as other topics can be more important. The size of the plantary balls is an indication of the topic's connectedness in the web.
- Leads - an explanation with full text for the nearest neighbours of the search topic.
- References - Actual document links or remarks made about the search topic.
- Same context - Other topics discussed in the same context.

[1] This is a general statement of the second law of thermodynamics – entropy tends to increase. If we want to reduce it locally, we have to expend a lot of effort.

Knowledge map example 1

You enter the knowledge map by searching, or being referred there by a link in the Mission Portal. Suppose we enter '`webserver`' into the search field. CFEngine searches for matching topics in its index, and returns references in different contexts. The context 'any' represents an unspecific reference to the topic name:


Mission Portal - Knowledge Map - Opera

cfengine.com/demo2/index.php/knowledge/knowledgeSearch

NOVA MISSION PORTAL

Hello insight | logout Online users: 1

MISSION PORTAL > LIBRARY

webserver

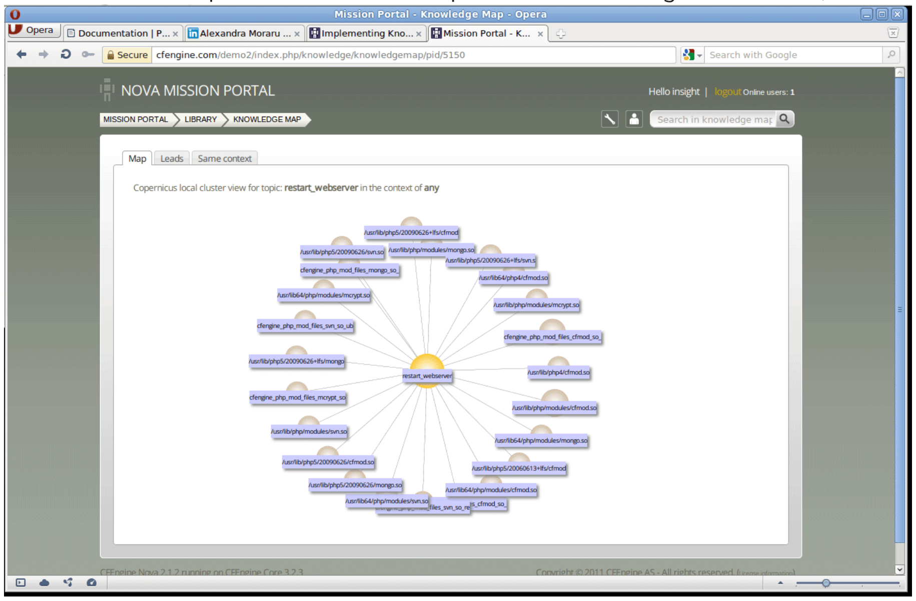
Results for search : webserver

"restart_webserver" is mentioned in the context of any
"restart_webserver" is mentioned in the context of class_contexts
"restart_webserver.debian" is mentioned in the context of class_contexts
"restart_webserver.debian" is mentioned in the context of any
"restart_webserver.redhat" is mentioned in the context of class_contexts
"restart_webserver.redhat" is mentioned in the context of any
"restart_webserver.suse" is mentioned in the context of class_contexts
"restart_webserver.suse" is mentioned in the context of any
"webserver" is mentioned in the context of any
"webserver" is mentioned in the context of application_services
"webserver in open nebula" is mentioned in the context of special_topics_guide
"webserver in open nebula" is mentioned in the context of any

CFEngine Nova 2.1.2 running on CFEngine Core 3.2.3

Copyright © 2011 CFEngine AS - All rights reserved.

Suppose we click on the first of these in any context; this leads us to a special landing page about that particular topic which contains the four tabs mentioned above. By default we end up in the graphical view. This gives an immediate impression of roughly where CFEngine considers this topic to lie in the scope of its knowledge. However, its value is rather limited.


Mission Portal - Knowledge Map - Opera

cfengine.com/demo2/index.php/knowledge/knowledgemap/pid/5150

NOVA MISSION PORTAL

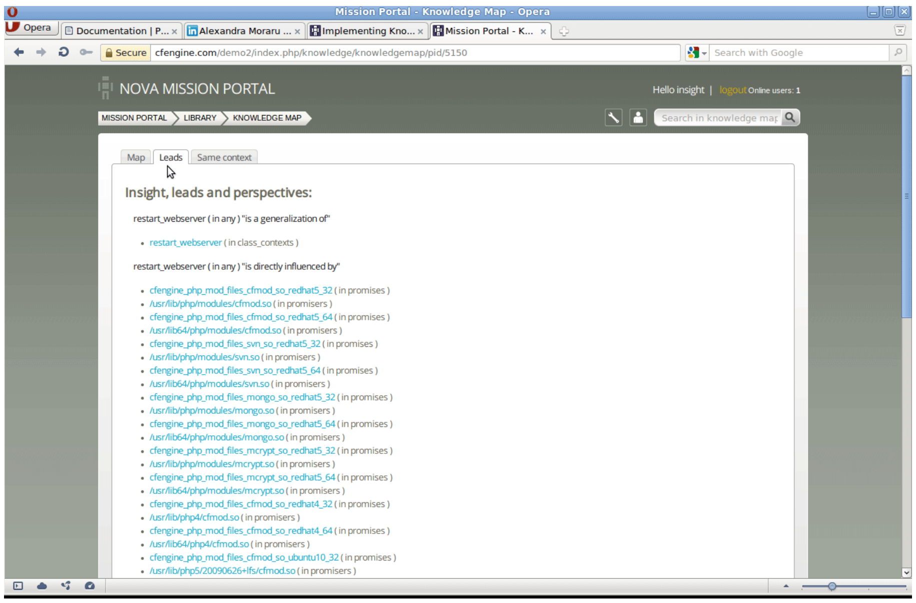
Hello insight | logout Online users: 1

MISSION PORTAL > LIBRARY > KNOWLEDGE MAP

Map | Leads | Same context

Copernicus local cluster view for topic: **restart_webserver** in the context of **any**
restart_webserver
/usr/lib/php5/20090626+lfs/cfmod
/usr/lib/php5/20090626/svn.so
/usr/lib/php/modules/mongo.so
/usr/lib/php5/20090626+lfs/svn.s
cfengine_php_mod_files_mongo_so_
/usr/lib64/php4/cfmod.so
/usr/lib64/php/modules/mcrypt.so
/usr/lib/php/modules/mcrypt.so
cfengine_php_mod_files_svn_so_ub
cfengine_php_mod_files_cfmod_so_
/usr/lib/php5/20090626+lfs/mongo
/usr/lib/php4/cfmod.so
cfengine_php_mod_files_mcrypt_so
/usr/lib/php/modules/cfmod.so
/usr/lib/php/modules/svn.so
/usr/lib64/php/modules/mongo.so
/usr/lib/php5/20090626/cfmod.so
/usr/lib/php5/20060613+lfs/cfmod
/usr/lib/php5/20090626/mongo.so
/usr/lib64/php/modules/cfmod.so
/usr/lib64/php/modules/svn.so

The next step is to walk through the tabs to 'drill down' to actual content. The second tab points us to the 'brainstorming' value of the associative links – now with explanations in text. If we select another tab we arrive at further leads for assisted thinking: these associative-thinking pointers can guide you to matters that are related and provide context for the current topic.



Insight, leads and perspectives:

restart_webserver (in any) "is a generalization of"

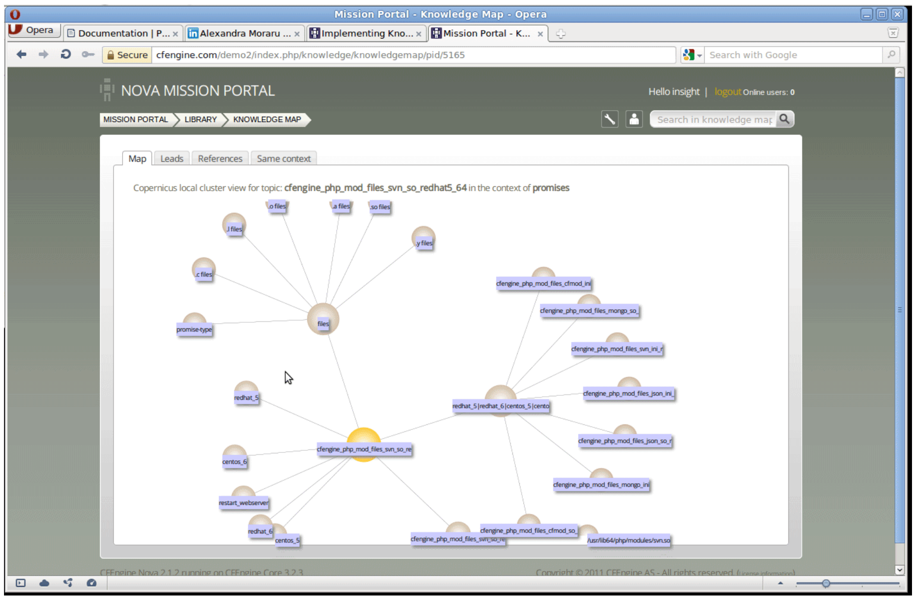
- restart_webserver (in class_contexts)

restart_webserver (in any) "is directly influenced by"

- cfengine_php_mod_files_cfmod_so_redhat5_32 (in promises)
- /usr/lib/php/modules/cfmod.so (in promisers)
- cfengine_php_mod_files_cfmod_so_redhat5_64 (in promises)
- /usr/lib64/php/modules/cfmod.so (in promisers)
- cfengine_php_mod_files_svn_so_redhat5_32 (in promises)
- /usr/lib/php/modules/svn.so (in promisers)
- cfengine_php_mod_files_svn_so_redhat5_64 (in promises)
- /usr/lib64/php/modules/svn.so (in promisers)
- cfengine_php_mod_files_mongo_so_redhat5_32 (in promises)
- /usr/lib/php/modules/mongo.so (in promisers)
- cfengine_php_mod_files_mongo_so_redhat5_64 (in promises)
- /usr/lib64/php/modules/mongo.so (in promisers)
- cfengine_php_mod_files_mcrypt_so_redhat5_32 (in promises)
- /usr/lib/php/modules/mcrypt.so (in promisers)
- cfengine_php_mod_files_mcrypt_so_redhat5_64 (in promises)
- /usr/lib64/php/modules/mcrypt.so (in promisers)
- cfengine_php_mod_files_cfmod_so_redhat4_32 (in promises)
- /usr/lib/php4/cfmod.so (in promisers)
- cfengine_php_mod_files_cfmod_so_redhat4_64 (in promises)
- /usr/lib64/php4/cfmod.so (in promisers)
- cfengine_php_mod_files_cfmod_so_ubuntu10_32 (in promises)
- /usr/lib/php5/20090626+lfs/cfmod.so (in promisers)

Leads are helpful if you didn't really know what you were looking for in the first place – and you would like some explanation of what it what you guessed.

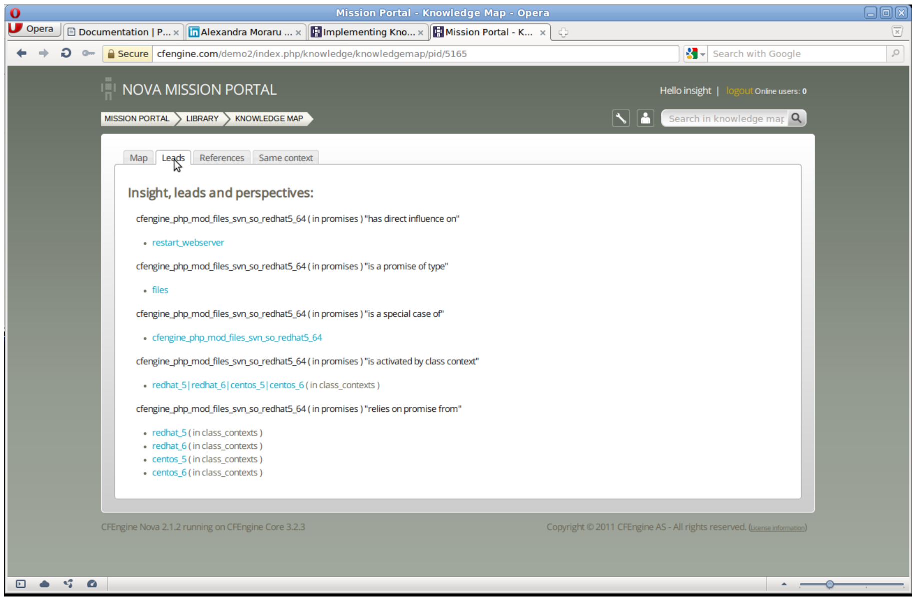
If we select one of these, we go the landing page for that new topic in the advertised context.


Mission Portal - Knowledge Map - Opera
Opera
Documentation | P...
Alexandra Moraru ...
Implementing Kno...
Mission Portal - K...
Secure
cfengine.com/demo2/index.php/knowledge/knowledgemap/pid/5165
Search with Google
NOVA MISSION PORTAL
Hello insight | logout Online users: 0
MISSION PORTAL
LIBRARY
KNOWLEDGE MAP
Search in knowledge map
Map
Leads
References
Same context
.o files
.a files
.so files
.l files
.y files
.c files
promise-type
files
cfengine_php_mod_files_cfmod_ini
cfengine_php_mod_files_mongo_so_
cfengine_php_mod_files_svn_ini_r
cfengine_php_mod_files_json_ini_
redhat_5
redhat_5|redhat_6|centos_5|cento
cfengine_php_mod_files_json_so_r
cfengine_php_mod_files_svn_so_re
centos_6
cfengine_php_mod_files_mongo_ini
restart_webserver
redhat_6
centos_5
cfengine_php_mod_files_svn_so_re
cfengine_php_mod_files_cfmod_so_
/usr/lib64/php/modules/svn.so
CFEngine Nova 2.1.2 running on CFEngine Core 3.2.3
Copyright © 2011 CFEngine AS - All rights reserved

Copernicus local cluster view for topic: cfengine_php_mod_files_svn_so_redhat5_64 in the context of promises

Note that several topics might be highlighted in yellow on a page – if they have been named, or if they are topics of special importance in this local cluster (meaning that they are highly referred to).

In this view we are able to see, at a glance, that this somewhat Byzantine string is the name of a file, and that is pertains only to certain operating systems. Walking through the tabs again, we see both where we came from and where we can go from here, with more detailed explanations.



Insight, leads and perspectives:

cfengine_php_mod_files_svn_so_redhat5_64 (in promises) "has direct influence on"

- restart_webserver

cfengine_php_mod_files_svn_so_redhat5_64 (in promises) "is a promise of type"

- files

cfengine_php_mod_files_svn_so_redhat5_64 (in promises) "is a special case of"

- cfengine_php_mod_files_svn_so_redhat5_64

cfengine_php_mod_files_svn_so_redhat5_64 (in promises) "is activated by class context"

- redhat_5|redhat_6|centos_5|centos_6 (in class_contexts)

cfengine_php_mod_files_svn_so_redhat5_64 (in promises) "relies on promise from"

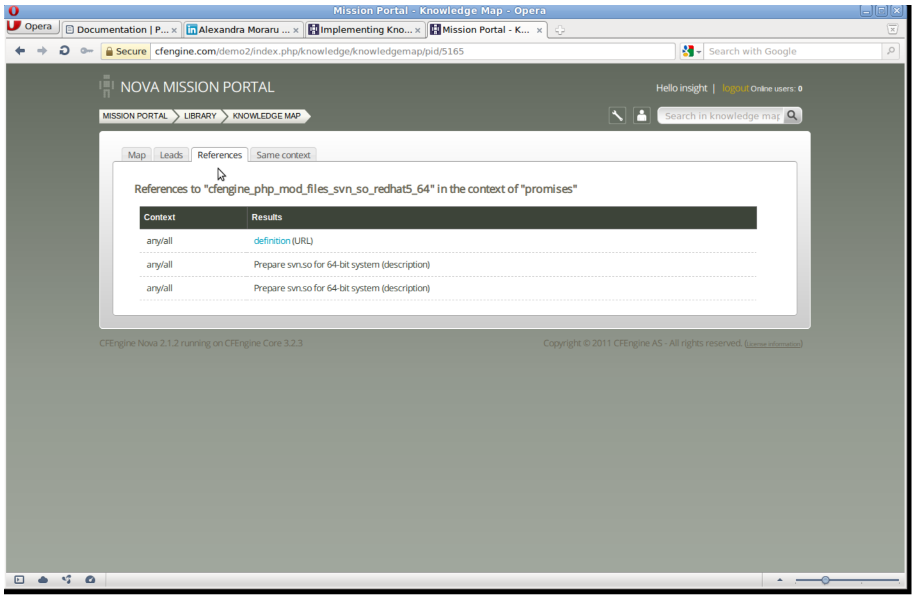
- redhat_5 (in class_contexts)
- redhat_6 (in class_contexts)
- centos_5 (in class_contexts)
- centos_6 (in class_contexts)

CFEngine Nova 2.1.2 running on CFEngine Core 3.2.3

Copyright © 2011 CFEngine AS - All rights reserved. (License information)

So far the map has only shown us topics – or named subjects categorized by their context. This gives us a kind of compass for knowing what the topic is about, but it tells us little about the subject itself. The third tab points us to actual expansive information about the topic, if any is known. This might be information from the documentation, external links or commentary and remarks made in policy.

If there are no references to external information, this tab does not appear. Let's choose one of these references; now we are taken to a new topic altogether, where we see a new view of the system information.



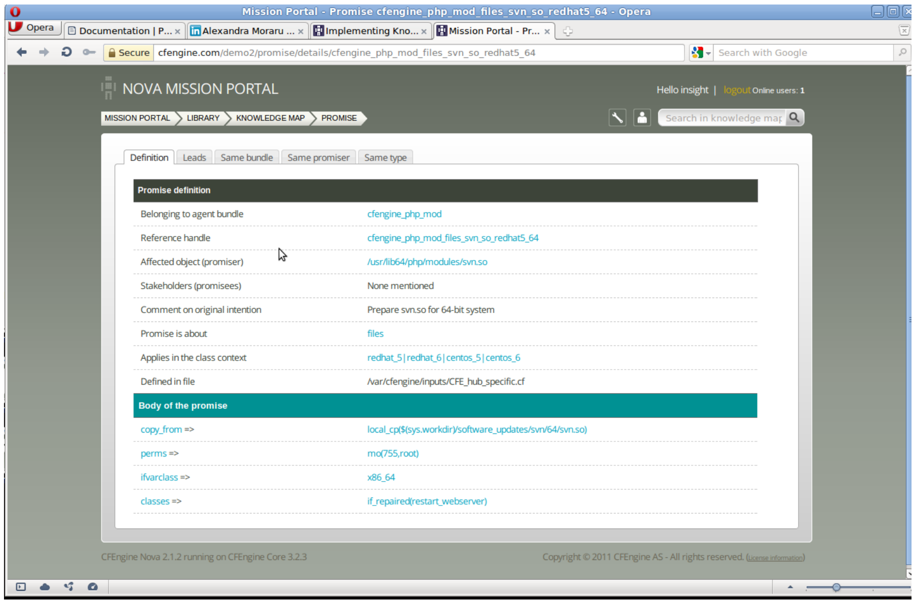
References to "cfengine_php_mod_files_svn_so_redhat5_64" in the context of "promises"

Context	Results
any/all	definition (URL)
any/all	Prepare svn.so for 64-bit system (description)
any/all	Prepare svn.so for 64-bit system (description)

CFEngine Nova 2.1.2 running on CFEngine Core 3.2.3

Copyright © 2011 CFEngine AS - All rights reserved. (License information)

Because the topic name is categorized in the context of 'promises', we know that (although this is the name of a file and it refers to RedHat-like operating systems) that this is also the object of a promise. The references page contains a link that then points us to actual information about this promise. The textual comment that was attributed to the promise in the policy itself, and a link to the promise's definiton. Clicking on the defintion link takes us the the promise-viewer.



Promise definition	
Belonging to agent bundle	cfengine_php_mod
Reference handle	cfengine_php_mod_files_svn_so_redhat5_64
Affected object (promiser)	/usr/lib64/php/modules/svn.so
Stakeholders (promisees)	None mentioned
Comment on original intention	Prepare svn.so for 64-bit system
Promise is about	files
Applies in the class context	redhat_5 | redhat_6 | centos_5 | centos_6
Defined in file	/var/cfengine/inputs/CFE_hub_specific.cf
Body of the promise	
copy_from =>	local_cp($(sys.workdir)/software_updates/svn/64/svn.so)
perms =>	mo(755,root)
ifvarclass =>	x86_64
classes =>	if_repaired(restart_webserver)

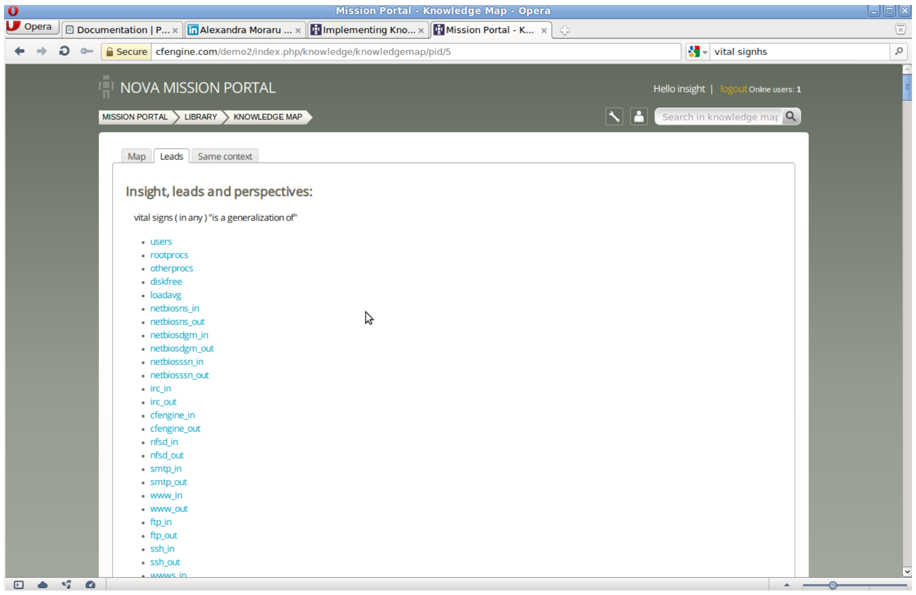
CFEngine Nova 2.1.2 running on CFEngine Core 3.2.3

Copyright © 2011 CFEngine AS - All rights reserved. (License information)

The Copernicus Knowledge Map view will take you a while to get used to – use it to help you think about matters, and see how they fit together into a systematic overview of your environment.

Knowledge map example 2

Let's look at another example. Suppose we enter '`vital signs`' as the search text, and go to the leads. We find a long list of other topics. CFEngine says that 'vital signs' generalizes, or is an umbrella concept, for these other concepts.


Mission Portal - Knowledge Map - Opera

cfengine.com/demo2/index.php/knowledge/knowledgemap/pid/5

NOVA MISSION PORTAL

MISSION PORTAL > LIBRARY > KNOWLEDGE MAP

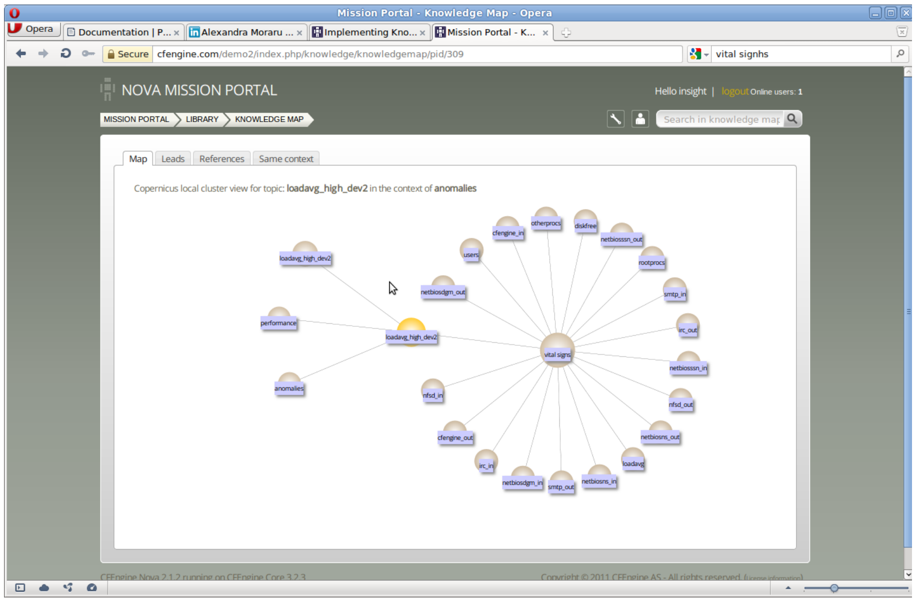
Map | Leads | Same context

Insight, leads and perspectives:

vital signs (in any) "is a generalization of"

- users
- rootprocs
- otherprocs
- diskfree
- loadavg
- netbiosns_in
- netbiosns_out
- netbiosdgm_in
- netbiosdgm_out
- netbiosssn_in
- netbiosssn_out
- irc_in
- irc_out
- cfengine_in
- cfengine_out
- nfsd_in
- nfsd_out
- smtp_in
- smtp_out
- www_in
- www_out
- ftp_in
- ftp_out
- ssh_in
- ssh_out

One of these concepts is a class context that represents an anomaly detected by CFEngine '`loadavg_high_dev2`'. Clicking on this, we see the significance of this concept quite easily from its relationship to other concepts:


Mission Portal - Knowledge Map - Opera

cfengine.com/demo2/index.php/knowledge/knowledgemap/pid/309

NOVA MISSION PORTAL

MISSION PORTAL > LIBRARY > KNOWLEDGE MAP

Map | Leads | References | Same context

Copernicus local cluster view for topic: **loadavg_high_dev2** in the context of **anomalies**

CFEngine Nova 2.1.2 running on CFEngine Core 3.2.3

Copyright © 2011 CFEngine AS - All rights reserved.

The load average being high might be an anomaly, but we also see that it refers to system performance, as does a whole bunch of other concepts.

The value of the semantic indexing is that it is information a casual user can actually learn from. By providing context and leads for further thinking, the Mission Portal becomes an quasi-intelligent assistant for working on the system.

Types of information

There are many kinds of information that contribute to a strategy for managing knowledge. The ability to learn from past mistakes demands that we look both forwards and backwards, while realizing that knowledge grows older and less useful the older it gets. For instance, what is known about the past? What is current? How can we track these things? Information comes in logically different categories as well as different types. Data have different sources, and are about different epochs. We need to unify and integrate these in Knowledge Management.

Conceptual information (Always)

Conceptual information is knowledge of a general nature that defines basic terms and concepts that describe the operational area of the organization. e.g. background theory, compliance requirements, frameworks, license expiry documents. Contracts and agreements.

This information comes from the wider culture of the enterprise. It is often treated poorly as our modern culture has an unreasonable suspicion of anything that seems 'academic'.

System observations (Past)

Recording what has actually happened in the organization. This includes the IT system, but also the business services and workflows, e.g. changes, transactions, logs, incidents, events, performance values (Key Performance Indicators), audits, security scans, etc. This includes what it traditionally called system monitoring.

This comes from probing and monitoring human-computer processes.

Enterprise process documentation (Now)

These documents are about business internals, recipes, manuals and how-tos. They allow individuals to work autonomously without constant supervision, because they all have a copy of the script. This information is about *orchestration* of parts. In an orchestra each player can manage to play his or her part with only minimal advice from a conductor because they each have a copy of their music. Their music is their process documentation.

This is closely related to policy and comes from internal management bodies.

Policy (Future)

Here are the promises and intentions of the IT mission. Documents of all degrees of technicality form policy. They contain the decisions you have made for steering the mission under all possible contingencies.

Policy is informed by all of the above, and usually changes in response to recurring 'problems' identified in the observations. It originates from internal management.

Stories and narratives

In the foregoing section, we saw how leads in the Knowledge Map's semantic index could help us to see immediate relationships between system issues. The next question is: what about connections that are less immediate or obvious?

In future releases you will be able to ask the agent to brainstorm for you about topics, telling any stories is knows about topics. A story is a simply a path through the web of connected information that follows some kind of connectness. Most interesting stories have a thread of cause and effect. CFEngine attempts to reason forth narratives that might be sensible or which might offer some kind of insight. This is not usually a directly useful revelation, but more often a thought provoking stimulus for a human to take further.

Machine intelligence is never a substitute for the real thing, at least not in our present day technologies, but it can be a useful amplifier of 'leads' to get a smart human thinking about the right things. Most CFEngine stories relate to policy, and they are generated automatically by the knowledge engine, so you will not get any stories until you have something interesting in your policy.

Stories will be usable from the command-line, or as a tab within the Mission Portal. The command line is useful for interactive thinking:

A story can be quite simple:

```
atlas$ src/cf-know --tell-me-about ftpusers
F: "ftpusers" seems to affect "any::/etc/ftpusers" (with 50 % probability)

  "ftpusers" (in any),
   see also "/etc/ftpusers" (in any),
```

Other stories could be more involved:

```
cf-know --tell-me-about anomalies::cpu0_high_dev2

F: "anomalies::cpu0_high_dev2"
seems to affect "any::performance" (with 100 % probability)

  "cpu0_high_dev2" (in anomalies),
   is a special case of "vital signs" (in any),
   which is a generalization of "users_high_dev1" (in anomalies),
   and this (users_high_dev1) is a special case of "performance" (in any)

(Note also that performance, which was mentioned, is a generalization of
"anomalies::users_high_dev1")
```

Stories can also be quite spurious, because knowledge itself can be spurious. For instance, some guesses might be quite wrong as the system gets the wrong end of the stick, so to speak, as it uses general patterns and these patterns always have exceptions:

```
atlas$ src/cf-know --tell-me-about operating_systems::suse

F: "operating_systems::suse" seems to affect "any::gnu/linux" (with 100 % probability)

  "suse" (in operating_systems),
   is a special case of "linux" (in any),
   and this (linux) seems to refer to "gnu/linux" (in any)

F: "operating_systems::suse" seems to affect "any::unitedlinux" (with 100 % probability)

  "suse" (in operating_systems),
   is a special case of "linux" (in any),
   and this (linux) seems to refer to "unitedlinux" (in any)
```

Or:

```
atlas$ src/cf-know --tell-me-about functions

F: "any::functions" seems to affect "any::arrays" (with 100 % probability)

  "functions" (in any),
   seems to refer to "functions which read" (in any),
   which seems to refer to "functions which read arrays" (in any),
   and this (functions which read arrays) seems to be referred to in "arrays" (in any)

F: "any::functions" seems to affect "any::expression" (with 100 % probability)

  "functions" (in any),
   seems to refer to "functions which read" (in any),
   which seems to refer to "functions which read classes" (in any),
   which seems to be referred to in "class" (in data_types),
   which seems to refer to "a cfengine class expression" (in any),
   and this (a cfengine class expression) seems to be referred to in "expression"
(in any)
...
```

As you can see, the story generator looks for causative connections between the things that it knows about. Some topics are from the documentation, and some from your configuration policy.

```
F: "any::functions" seems to affect "any::readrealarray" (with 100 % probability)

  "functions" (in any),
   seems to refer to "functions which read" (in any),
   which seems to be referred to in "class" (in data_types),
   which seems to be referred to in "functions which return" (in any),
   which seems to refer to "functions which return real" (in any),
   which seems to be referred to in "real" (in vars_promises),
   and this (real) seems to refer to "readrealarray" (in any)

(Note also that readrealarray, which was mentioned, seems to refer to
"any::functions which read classes")
(Note also that readrealarray, which was mentioned, seems to refer to
"any::functions which return class")
```

Creating your own map

CFEngine provides a lot of knowledge about your system out of the box, that includes a domain model about the space of IT management, and an automated analysis of the deployed policy. Below are some notes about how this is built, and things you can do to improve the information by adding knowledge that cannot be discovered automatically.

1. You begin by documenting your intentions by creating a CFEngine automation policy, i.e. bundles of promises.
2. You add comments, handles, promisees etc to your policy to full explain your intentions.
3. You can assign each host in your network to a special class that represents its physical location, using `classes:` promises. You should then collect such classes into a topic context called '`locations::`' in the '`company_knowledge.cf`' file.
4. CFEngine runs `cf-promises -r` to build a decompsition of your current policy.
5. CFEngine runs `cf-know -bf enterprise_build.cf` to build the knowledge map.
6. You may create enterprise process documents and written policies for your organization, which you will link into the knowledge map.
7. CFEngine Nova creates the Knowledge Map which includes a conceptual framework for IT management, and which integrates with the CFEngine software documentation.
8. You may now extend the basic knowledge map with more of your own special documentation by placing references and additional concepts into the file '`company_knowledge.cf`', on the policy server.

Documenting goals

Business goals are shown in the Goals app (and service catalogue) in the Mission Portal of CFEngine Enterprise for each hub. To document the fact that a promise or bundle contributes to a business goal, you must do two things:

1. Document the goal (see below under the example '`company_knowledge.cf`' file). e.g.

```
topics:

  goals::
    "goal 1" comment => "Do good things";
    "goal 2" comment => "Be first";
    "goal 3" comment => "Be best";
```

2. Make the goal a promisee of the promise concerned.

```
methods:

   "security"  -> { goal_1, goal_2 }

                   comment => "Basic change management",
                 usebundle => change_management;
   "maintenance"
                   comment => "Perform log rotation",
                 usebundle => garbage_collection;
```

Documenting locations

To document the location of a host, make sure that it is placed inside a class that represents its address:

```
classes:

  "london"     or => { "host1", "host2" };
  "alexandria" or => { "host3", "host4", "host5" };
```

Then designate these classes as locations in the knowledge map:

```
topics:

 locations::

  "london"     comment => "29 Market Street, London XW4, third on the left";
  "alexandria" comment => "Secret chamber, discovered by Indiana Jones";
```

Example ‘company_knowledge.cf’

```
bundle knowledge company_knowledge
{
things:

 regions::

   "Americas" comment => "USA, Latin America and Canada";

   "EMEA"     comment => "Europe, The Middle-East and Africa";

   "APAC"     comment => "Asia and the Pacific countries";

 countries::

   "USA";

   "Germany";

   "UK"            synonyms => { "Great Britain" },
              is_located_in => { "EMEA", "Europe" };

   "Netherlands"   synonyms => { "Holland" },
              is_located_in => { "EMEA", "Europe" };

   "Singapore"     is_located_in => { "APAC", "Asia" };

 site_locations::

   # Use this for the names of

   "London_1"  is_located_in => { "London", "UK" };
   "New_Jersey"  is_located_in => { "USA" };
 routers::

  "oslo-hub-p6 "   comment => "Cisco xyz router, 3rd floor machine room, 6 Penny Street";
  "oslo-hub-trunk" comment => "Cisco BGP router,  floor machine room";
  "nyc-hub-456"    comment => "Juniper 123 router, 3rd floor machine room";

 networks::

  "192.23.45.0/24"
                  comment => "Secure network, zone 0. Single octet for corporate offices",
          is_connected_to => { "oslo-hub-123" };

  "192.12.74.0/23"    comment => "Zone 1, double octet for the London office developer network",
              is_connected_to => { "oslo-hub-123" };

  "192.12.74.0/23"
                     comment => "Secure, single octet for the NYC office",
             is_connected_to => { "nyc-hub-456" };


#######

topics:

 company::

  "ED"  comment => "Exceptional Devices Ltd";

 ED::

  "EGC"             comment => "Enterprise grid computing",
                association => a("develops stuff within",
                                 "ED",
                                 "contains its engineering unit");

  "EBC"             comment => "Enterprise business computing",
                association => a("handles business services within",
                                 "ED",
                                 "contains its business unit");

  "ED terminology" comment => "Internal company nomenclature";

 ED_terminology::

  "interactive job"   comment => "Interactive software running in the grid";

  "apps"              comment => "Software applications",
                  association => a("are also referred to as",
                                   "interactive jobs",
                                   "are also referred to as");

  "business services" comment => "Support services for sales";

########################
# GOALS as topics
########################

  goals::
    "goal 1" comment => "The company mission depends on reliability to our customers";
    "goal 2" comment => "Should be running recent versions of key software";
    "goal 3" comment => "The company must be compliant with Sarbanes Oxley act.";
    "goal 4" comment => "Comply with US Export restrictions for class D countries";


#######
# Documents below
#######

occurrences:

 # Fill in company references here

 work_shifts::

    "http://www.example.com/shifts_and_rotas.ods"
       represents => { "Spreadsheet" };


 current_projects::

    "http://www.example.com/scope_of_work.html"
       represents => { "SOW" };


 software_licenses::

    "This CFEngine software license expires <b>$(sys.expires)</b>"
   representation => "literal",
       represents => { "CFEngine Nova" };

}
```

What other special documents should an organization have?

CFEngine cannot produce everything you need for your library of knowledge. There cannot be hard and fast rules about this. Here are some suggestions:

- A policy on human roles and responsibilities, to avoid gaps in and unecessary doubling of effort.
- A risk/security policy.
- A standardization and procedure manual.
- Incident logs of matters not detected and repaired by CFEngine.
- Repair logs of matters not detected and repaired by CFEngine.
- Requests for change (RFC) logs.
- Promises in your organization that are not just for CFEngine to keep, documenting intentions.
- A policy on Knowledge Retention and Business Continuity.

Hint: a simple way to have a log is to use a tickets system like OTRS or a moderated mailing list, and to email new entries. Something like a Mailman archive allows then tracking and integration.

Knowledge transfer

Passing on knowledge to others could be considered a 'best practice'. Alas, this is not as easy as it sounds. There are plenty of strategies to achieve knowledge transfer:

- Reading
- Taking courses
- Research
- Repetition
- Job swapping and promotion

These are just a few. However, this list is old news, and these items alone will not lead to knowledge transfer.

One of the most important *barriers* to knowledge transfer is that individuals refuse to learn new things. A lack of dynamism in a company or organization establishes patterns of habit that are hard to break. The longer they last, the more likely they are to continue.

Promise theory makes clear that, even if all staff promise to write excellent documentation, they do not necessarily promise to read each others' documents. Transfer requires a commitment both to send and to receive.

- The final step to managing knowledge is to foster a culture of knowledge acquisition, retention and transfer.
- Knowledge transfer requires a commitment by all parties
  - to learn outside of their box.
  - to pass on their expertise to others.

Communication skills are clearly important, but this is a serious flaw in the idea. Written communication skills are not as common as we might think. This means it is hard for the writer, and also for the reader.

The solution taken by CFEngine is to reduce the problem of documentation to one of coding notes and relationships. This requires only minor writing skills. The technology can then assemble these notes using intelligent algorithms and present the result in an organized form with visual aids. You should pay special attention to the syntax items:

```
comment =>
handle =>
depends_on =>

-> { promisees }
```

You need to foster a culture of using information, in order for it to become assimilated as knowledge. Unused information will perish.

How does CFEngine Nova help?

- CFEngine assists knowledge management by providing tools for integration of knowledge sources.
- A formal structure for encoding policy at relationships at a functional level
- Technical syntax makes annotation easy for IT workers, and CFEngine can construct the surrounding story using intelligent algorithms.

Knowledge Management Objectives

As a manifesto for us as developers at CFEngine, as well as for you as users of the software, and as infrastructure engineers, it is helpful to make a list of questions that we would like to be able to answer using enterprise software. The following list is not to be regarded as a set of features in CFEngine, but rather as a set of challenges to be addressed in different ways.

- Tell me about the promise(s).
- Tell me what happened to Y.
- Tell me about security.
- Tell me about performance.
- Tell me about compliance.
- Where are the resource problems?
- Where is the greatest/least activity?
- How much spare capacity do I have?
- Where are things changing fastest?
- When do I need to think about X?
- What do I need to know about X?
- What are the most important things that happened?
- Why is this here?
- What can I do about X?
- Who changed X last?
- When can I expect X?
- Where can I find X?
- What things affect X?
- What promises have been made about X?
- What stories lead to conclusion X?