

The Information Management Model

By Rob Hanna

Our grasp of single-sourcing has come a long way in the past few years. This is thanks in part to technology that makes it easier to reuse content and in part to our pundits that introduce new ideas into our community.

However the practice of single-sourcing is not new. For decades other industries, such as manufacturing and software engineering, have been producing components designed to be reused in products across their companies and their industries.

What we lack that has made single-sourcing successful in other domains is a common standard for the components.

To reach any real measure of success, we must seek to standardize how we manage information. The Information Management Model is an idea that aims to take a step in that direction.

SINGLE-SOURCING INFORMATION

Our role as technical communicators within an organization is straight forward. We capture information provided by subject matter experts as it changes. We then transform it into content and publish it to our users. Yet when we talk about single-sourcing, most often we are talking about managing our own content. If we were able to single source information rather than content we could get closer to the source and interact directly with the agents of change within the organization. The challenge is in finding a way to effectively manage information as content.

Separating the Information and Presentation Layers

One of the hardest concepts to grasp when employing single-sourcing is the true separation of the information layer from the presentation layer. We understand the need to separate format from content to successfully single-source but we should be thinking of separating information from content if we want to be truly successful.

When we write, it is difficult to do so without consideration for how it is going to appear in the finished product. We tend to think of information models based upon the structure of our information products. Accepted practice suggests that we begin planning our models based upon content produced for various documents and online products. This is so that we can identify similar or

identical content that can be consolidated into a single source.

Language, format, media, audience, and even derivative content are all part of the presentation layer. They can all change according to the needs of our users. The underlying information however stays the same regardless of changes to the presentation layer.

When we think of content within a content management system, we think of fragments of documents that sit within a repository that can be reused at will in other documents. The problem that I see with this is that we are still focused upon the presentation layer any time we discuss content

The term content itself implies that it is contained in something. The container is always the presentation layer. If we change the container, we need to change the content, unless we have carefully crafted the content to fit in other anticipated containers without modification.

If we do not know what containers will be used to house our content, we must rely partly on chance that the content will work appropriately in any new context. This is sometimes referred to as accidental reuse. Experts argue that accidental reuse is not practical – and this I would agree is mostly true of content. But it is not strictly true of information. Information should not change according to its context.

The Information Management Model

I have endeavored to create a model that attempts to define the shape of information we use to create documentation in a single-sourcing environment. The model has a basis in many different architectures we may have encountered (some are described later on in this paper). It addresses the issue of content granularity by defining the types of information that we will be capturing and reusing. It is strictly linked to the lifecycle of information as it is created and modified throughout the product development lifecycle. It is process driven and puts ownership of the information into the hands of the true agent of change within an organization giving us access to the actual source of information we will use to publish. The model does not restrict the format or function of our information products but will instead improve upon how we gather and verify the information we use.

Information is gathered and organized into Information Objects. These objects are controlled inside a repository representing the definitive source of information for a given environment. New information products are written and published incorporating information objects contained within the repository (see Figure 1).

Information Objects

Information objects represent separate units of meaning within the single-source environment. Each object is defined within a framework that describes each type of information to be managed within the single-source environment. This framework defines 11 primary information types and depicts the relationships between the objects. The framework is generic enough that it can be easily visualized and committed to memory and yet it is specific enough to provide adequate distinction between different information types.

preceding node within the framework. It is this traceability that sets the entire environment in motion. As information changes, it prompts for changes amongst sibling or children objects that may exist within the framework.

Traceability is a common feature within most process driven documentation environments. Within a software development lifecycle, requirements are stated that describe how a new product or feature should work. Requirements are used to create design elements. Design elements then are used to build the product upon which specifications are written. The requirements are the parent node. The designs are a child node. And the specifications are a child of the design node. The specs and designs can be changed as long as they are within the parameters of the stated requirements. But changing the requirements will necessitate the change within the children.

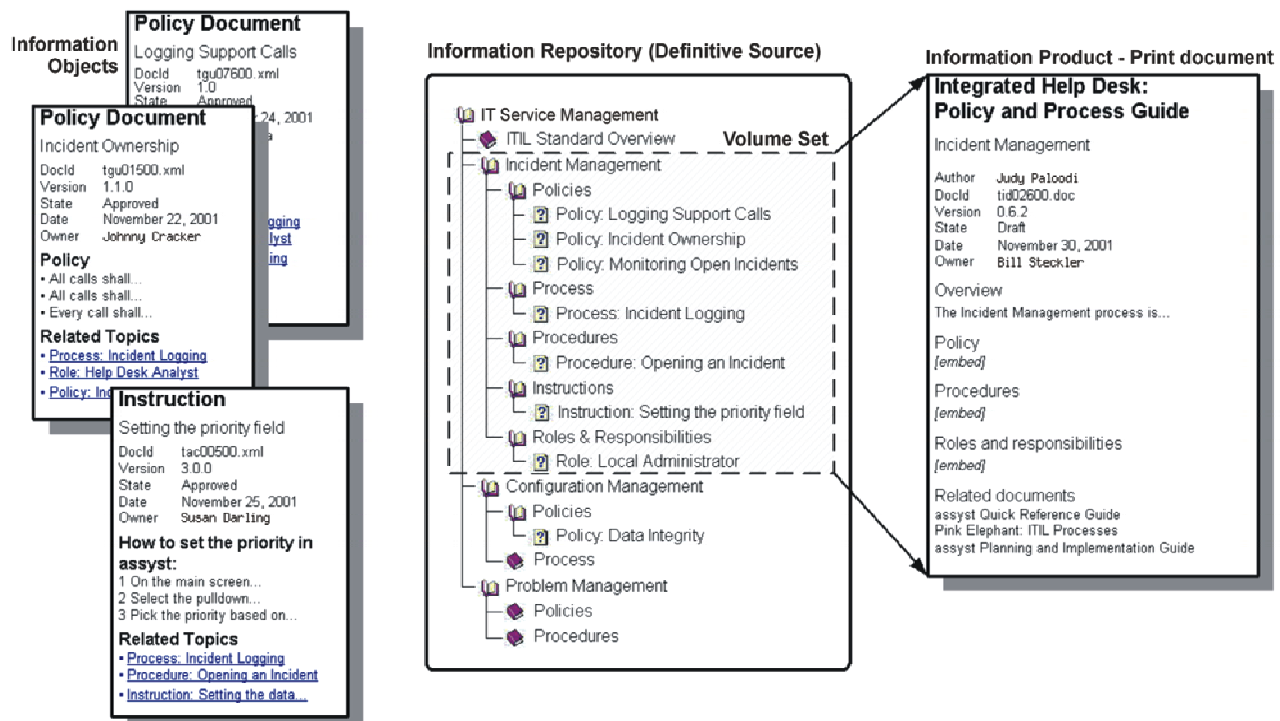


Figure 1. The model consists of information objects controlled within a central repository. These objects are used to create any number of information products.

Natural Traceability

The natural traceability between information types is one of the features that makes this model different from others I have seen. Each object has a natural dependency upon another object found within the same primary object type classification, or node, or upon an object in a

Object Specialization

While only 11 primary object types have been defined, there can be any number of object types that will be derived from the primary object types. An action object type, for instance, will be specialized to create processes, procedures, and instructions object types. Procedures can

be specialized as needed to create any number of subtypes.

The concept of specialization is employed in DITA (Darwin Information Typing Architecture) to allow for new information types to be derived from one of the four defined DITA information types (Topic, Task, Concept, and Reference). These derivative types share enough common structure with the basic type that they can be reduced to its most basic form if it is used in different environments. This facilitates universal reuse anywhere DITA is in place.

Progressive Disclosure

One of the main purposes of the information object is to capture all information pertaining to a specific unit of meaning within a single object so that if the information needs to be changed, it will change the object in each context in which it is used. Each object has a label which represents the object in a sentence.

For example, the concept of configuration management can be represented simply by its label “configuration management” in a sentence, by its associated acronym “CM” in an acronym list, by its label and short description within a glossary of terms, and by its label and full content as a chapter describing configuration management.

Each instance described above would point back to the same object. This could be used to verify spelling, usage and definition for the unit of meaning. Therefore we do not want to have separate objects for the same unit of meaning as it may be used in different contexts. We instead must be able to specify what elements within the object are reused within any given information product.

Nesting of Objects

An interesting property of any of the primary object types is that they can all be subdivided into smaller units as necessary. Any requirement can be broken down into more specific requirements. A procedure breaks down into subprocedures that may be represented as steps. An instruction can be nested inside a procedure that is nested inside a process—each identified as an action object type. This nesting creates another type of top-down traceability of objects where children are impacted by changes to the parent object.

Volume Sets

Objects are added to the environment as they are written and do not necessarily require that other objects preexist. As objects are linked within the environment according to their natural traceability paths, they begin to form volumes of information. A volume set refers to any contiguous set of objects within the environment. A complete volume would have linking that could trace from any one object within the volume set to another up to the highest node in the environment.

Scope and Ownership

Scope and ownership are defined for every object within a given environment. Scope determines the applicability of the information within the environment. Scope can be limited to a single role as described within the environment or it can apply to the entire organization. For example, a term may be defined within the scope of a department that is defined differently in another department. Scope allows both definitions to coexist within the same environment. Ownership is always defined for a single role as defined within the environment. The owner is responsible for the object.

THE MODEL FRAMEWORK

The model framework shows the relationships between the 11 different primary object types. The 11 types are the result of attempts to simplify and group object types that share similar construction. The relationships describe the natural traceability of the objects (see Figure 2).

Not only is it interesting that the resulting structure is almost symmetrical, its very orientation conveys additional meaning. In the vertical, the top of the model is expressed in the future tense for objectives; the middle of the model is expressed in present tense; and the tail of the model is expressed in past tense to describe a result of something that has already happened. In the horizontal, the left side of the model represents an organization’s human capital or service side; the center represents an organization’s intellectual capital or know-how; and the right side represents an organization’s physical capital or products.

These characteristics make for remembering the model as well as using it to categorize information much easier. The arrowhead shape also made for a good slide presentation to management showing business goals at the apex of all information within our domain.

The Primary Information Types

The 11 primary types are defined as follows:

1 Objective

The Objectives object type describes why a particular volume of information exists within the environment. It is the parent node that all other objects may eventually trace to. As objectives change, the entire body of information changes.

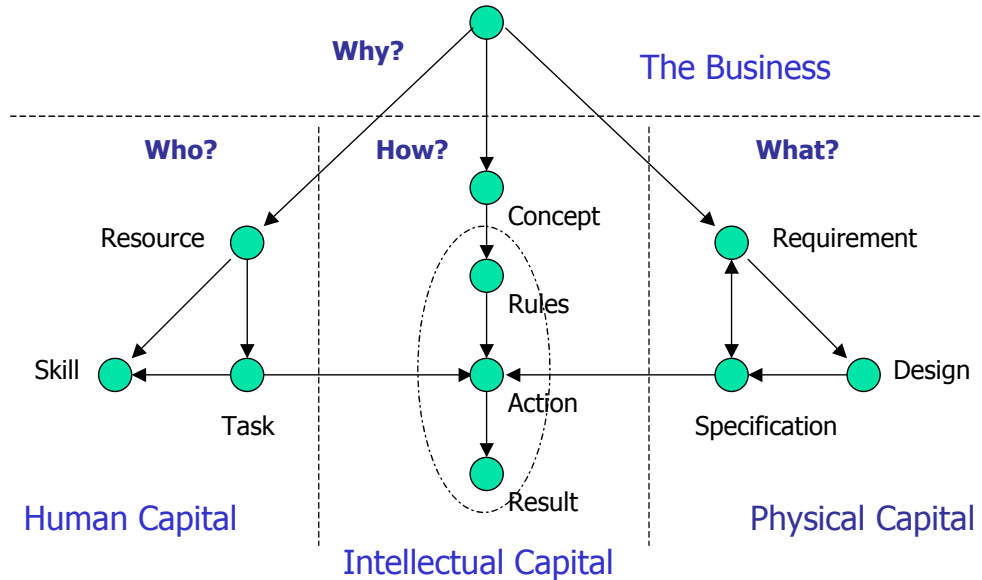


Figure 2: The framework of the model shows the 11 primary information object types, or nodes. The arrows represent the dependencies between the objects.

Objectives define specific challenges along with a timeframe and criteria to measure successful completion of objectives. Timeframe may be finite expressed as days, months or years or it may be open ended to describe any time from the present forward.

The highest node within an environment is an object that describes the mission of the organization. If the environment pertains to a department than the highest node is that department's mission statement. Any volume set that traces to this node is known as a complete volume.

Objective objects may be used to construct business plans and project charters.

2 Resource

Resource objects identify individuals and groups of people that act as a single entity within an environment. A resource can describe an actual person or group or it

can describe roles and cross-functional teams within an environment.

A person object will belong to one or more role objects. Role objects will belong to teams or groups. Teams or groups will belong to departments and so on until the entire organization within the environment is described. The hierarchy of these objects is significant and describes the structure of the organization represented in the environment.

Every information object is linked to two or more

resource objects that will describe the scope or domain of the object and also assign ownership to a single role within the environment. For example, a policy may be written that applies to a group within a department. The policy has no application outside of the group, therefore the resource object defining the group defines the scope of the policy object and is linked to that object. The policy requires an owner and is therefore linked to a role object that represents the role

of team lead for that group.

3 Task

Task objects describe what a given resource is expected to do. It describes tasks that the different roles within the environment are responsible for. Task objects also describe services that different groups within the environment offer. Service levels and criteria used to measure performance are described here. Tasks link users to procedures as well as requisite skills.

4 Skill

Skill objects describe levels of proficiency or prerequisites needed to perform tasks. Role and person objects (Resources) are linked to levels of proficiency they either require or possess. Task objects also link to levels of proficiency to describe what is needed to complete a task.

5 Concept

The concepts object defines the complete set of terminology used within the environment. Terms defined here can be used throughout the environment.

6 Rule

Rule objects describe policy or guidelines that shape how actions are performed. They provide warnings, cautions and notes that are highlighted while performing actions.

7 Action

The actions object sits at the heart of the model. It is the single node that links people with products. An action object describes how a task is to be performed by way of one or more steps. Actions objects include processes, procedures, instructions, and test plans.

8 Result

The results object sits at the base of the model linked to action objects. It represents the result of an action at a specific point of time in the past. As such it is written in the past tense. This object includes analysis of the result. While it appears as a terminal node in the model, it can ultimately link back directly to a new objectives object. Examples of results objects include scenarios, test results, and incident descriptions.

9 Requirement

The requirements, design and specification objects are all similar in that they can all describe a single item (system, module, tool, document) within the environment. Yet they are distinct in the way they are handled within the model. The highest level node of the three is the requirements object. This object describes how an item is expected to function. It is not intended describe how an item does function (even though the two may be similar if not identical). There may often be gaps between what is needed and what is provided. This information is preserved within the model. Examples of requirements objects include system requirements, requirement change, change requests, and data item definitions.

10 Design

The design object describes how the item will work or how requirements can be met. This object type encompasses a broad range of formats within the environment. It can include information conveyed in templates, source code, schematics, and flowcharts.

11 Specification

The specifications object describes the fit, function and finish of an item within the environment. Anything used by a person to perform an action is described as a specification. This can include descriptions of tools, interfaces, hardware, and documentation.

The Information Management Model and Mil-Std 2167/498 (U.S. DoD)

None of these concepts are essentially new at all. In fact, my earliest exposure to this was while working with CAE Electronics Ltd (now CAE Inc.) on maintaining software documentation that was first engineered in accordance with the U.S. Department of Defense standard 2167. At CAE, we broke down our Software Requirements Specification (SRS) and Software Design Document (SDD) into individual “paragraphs” or configuration items (CIs) that corresponded with units of code or functionality. Each CI was managed within the SDLC along with code to be integrated into the next release of the software.

All traceability between testing documents, incident reports, requirements, specifications, and design documents were managed within a database. All CIs were subject to the strictest of configuration management practices. As I was also responsible for their intranet, I managed the department’s policies, procedures and standards in a similar manner.

The Information Management Model and ITIL (IT Infrastructure Library)

The next major influence to the development of the model came while working at Canadian Tire in their IT division. A major part of my responsibilities at the time was to examine the way in which our Information Development group created documentation to support software maintenance activities.

The division had in place four separate help desks along with several support teams, development and QA functions that used disparate systems that did not integrate well. The decision was made to adopt the IT Infrastructure Library (ITIL) framework and find a single ITIL-compliant tool that could support the different areas across the SDLC.

As I began working with ITIL and documenting these new processes for managing our ITIL-compliant tool, I started to recognize patterns familiar to my model. ITIL breaks down IT service and infrastructure into configuration items (CIs) that describe how the IT

organization functions. The practices for configuration management, change control, and application development can serve as best practices for managing information objects just as it does managing any other CI in the system.

CONCLUSION

Still, at this point, the model is mostly theoretical. Since I began conceptualizing the model in 2002, I have been able to employ some of the information types I defined when writing new topics. I have, however, lacked the tools needed to fully realize the model. I believe that DITA holds the key to the deployment of a working model. My next steps include developing DITA specializations for the information objects and prototyping the model in a CMS.

There is much more to this model than can be described in this whitepaper. There are obvious parallels to other information models that are only eluded to here, such as DITA and Information Mapping®. As well, the specific design of the information objects is not defined in this document. The workflow and management of the objects is also critical to the success of this single-sourcing strategy.

It is my hope that this paper will stimulate some discussion on the merits of a standardized information model. While this model may not be the definitive model that our we choose to standardize on, it is my belief that we will seek and accept some form of universal model for information.

REFERENCES

- (1) Single Sourcing: Building Modular Documentation; Kurt Ament 2003, William Andrew Publishing.
- (2) Managing Enterprise Content: A Unified Content Strategy; Rockley, Kostur, Manning 2003, New Riders
- (3) An Introductory Overview of ITIL; Colin Rudd 2004; The IT Service Management Forum (ITSMF)
- (4) Introduction to the Darwin Information Typing Architecture; <http://www-106.ibm.com/developerworks/xml/library/x-dita1/index.html>

Rob Hanna

Manager
Toronto STC Single-Sourcing SIG
168 Bogert Ave
Toronto, ON M2N 1K9 Canada
Tel.: (416) 723-4183

Rob Hanna is a Senior Technical Writer at International Financial Data Services (IFDS) where he is responsible for user documentation for their suite of software solutions for the financial services sector.

Hanna has been writing professionally across several industries since 1990. He has also enjoyed writing for many different types of audiences. Starting from a background in aviation, his career lead from writing reports and operation guides for municipal airports to the role of production editor for one of Canada's largest circulating aviation periodicals.

In 1997 his career began to take more of a focus on writing than aviation as he stepped off into the role of technical writer for CAE Electronics. Soon software engineering began to play a more prevalent role in the types of documents Hanna was responsible for. Since leaving CAE in 2001, Hanna has worked for two other information technology groups at Entrust Technologies and Canadian Tire before joining IFDS in Toronto.

Hanna is a Senior Member of the Society for Technical Communications and a very active participant in the local technical writing community. In 2002, Hanna was instrumental in forming the Toronto STC Single-Sourcing SIG (Special Interest Group). Since then, the SIG has grown from a small clutch to a mailing list in excess of 70 members. Hanna is now the Vice-President of the Toronto STC Chapter.

For more information, you can review Hanna's career profile and portfolio online at <http://www.ascan.ca/info>. You can reach him at rob.hanna@ascan.ca.