CONCEPT CELL MODEL FOR KNOWLEDGE REPRESENTATION

JEFFREY Z. J. ZHENG*
Department of Information Security, School of Software
Yunnan University, Kunming, Yunnan, P. R. China
Conjugate Systems (Kunming) Limited Company
Kunming Yunnan, P. R. China

CHRISTIAN H. H. ZHENG
Conjugate Systems Pty Ltd
45 Greenways Rd, Glen Waverley, Victoria, 3150, Australia

TOSIYASU L. KUNII
Department of Computer and Information Engineering
Graduate School of Engineering, Kanazawa Institute of Technology
1-15-13 Jingumae, Shibuya-ku, Tokyo 150-0001, Japan

Received 6 January 2004
Revised 8 April 2004
Accepted 24 April 2004

This paper proposes a unified knowledge model establishing a core structure of knowledge representation in natural and artificial intelligence systems. The Concept Cell Model proposes the use of acyclic lattices to model a concept formed from a knowledge network of simpler concepts. Declarative and procedural knowledge are explicitly defined as the time-invariant and time-variant relationship of concepts. Examples of a Restaurant Servicing Concept Cell and an extended Shopping Complex Concept Cell are used to demonstrate the functionality of this model. Major existing theoretic and engineering Ontology knowledge schools are compared under this framework.

Keywords: Knowledge model; ontology; concept cell; knowledge representation.

1. Introduction

Knowledge modeling will play an essential role in any system requiring the use of intuition. Businesses have long relied upon humans as a vital cog in their operations, hence any innovation that could automate the process of intuition, let alone human intuition, would be regarded as revolutionary. The development of artificial intelligence technologies has seen a

*Corresponding author. E-mail: knowledgemodel@yahoo.com
need for organised management systems with the means of modeling concepts so that mechanical processes could be used to model the human processes such as ‘learning’ and ‘justifying’ — processes that could be seen as the implementation of knowledge.

Knowledge, with its abstract and constantly evolving nature, have confounded philosophers and knowledge engineers alike for centuries. Although the philosophical nature of knowledge can be debated with no definite answers, the engineering processes involved in the representation of knowledge (known as ontology) have been studied and developed for a long time. Throughout the knowledge engineering world, various schools of thought have been established, each presenting a case for knowledge in their own subjective view.

These schools have achieved many milestones with the establishment of numerous models and applications in relation to the progress of ontology. Yet all of them have various conflicting views with each other; all these schools have held beliefs that are contradicting to one another’s representation of knowledge and this in turn have stagnated the further development of the ontology process. It is clear at this point in time that a unified model of all existing ontology schools need to be proposed. This paper puts forward a unified model of all schools of thought, resolving the contradictions by modeling knowledge based upon an acyclic lattice network, constructing a solid foundation for modeling complex theoretical structures.

This paper proposes the use of basic building blocks in the form of Concept Cells to model and represent knowledge. The idea of the Concept Cell Model expands upon the notion of Classes, formulating an integrated framework to model any knowledge system. The focus of this paper is to establish knowledge as a layered structure with time-invariant and time-variant properties. Through the two synchronized structures, a concept can be consistently and precisely expressed and also applied to large scale knowledge structures. A small example of the flexibility of the model is given in Sec. 4 to model the services provided by a restaurant and by a shopping complex. In addition, a comparison of existing ontology knowledge schemes with this model has been listed in Sec. 5.

1.1. Summary of existing knowledge models

Whilst philosophers, from Plato to Descartes, have been debating the basis of knowledge for thousands of years with no apparent philosophical solution, executable solutions of knowledge have been modeled and refined since the 1940s starting with Michael Polanyi. Polanyi’s proposal [Polanyi, 1969, 1997] was that knowledge could be distinguished into two categories:

**Tacit Knowledge**: Knowledge that cannot be articulated but can be transferred to others through other means. Such knowledge includes personal relationships, daily experiences, moral beliefs and facial recognition.

**Explicit Knowledge**: Knowledge that is transparent and therefore able to be directly communicated to others. Such knowledge includes compiled data, formal procedures and policies.

Another approach, proposed by John Anderson [Anderson, 1976, 1993, 1995], coming from a cognitive psychology background, categorized knowledge into two types:

**Declarative Knowledge**: Very similar to explicit knowledge in that it explicitly describes facts, methods and procedures. Anderson viewed Explicit Knowledge and Declarative Knowledge as synonymous.

**Procedural Knowledge**: Viewed by Anderson as knowledge that is intrinsic and gained in the act of doing. Mental and physical skills such as playing the piano and dancing are types of Procedural Knowledge.

The unification of Polanyi and Anderson’s models began with Ikujiro Nonaka’s knowledge cycle [Nonaka, 1991, 1995]. Nonaka concentrated on the transformation and the communication of knowledge, addressing four transfers of knowledge using the Polanyi scheme. The cycle can be seen in Fig. 1. Each of the transfers are explained in detail.
Concept Cell Model for Knowledge Representation

Fig. 1. Nonaka’s tacit-explicit knowledge cycle.

Socialization (Tacit to Tacit) is the acquisition of knowledge through observation, imitation and practise. An example of this type of knowledge transfer is of a chef’s apprentice who learns from a sushi master by observing the master’s actions. Through this process, the apprentice could even develop his own technique and surpass the master.

Externalization (Tacit to Explicit) is the implementation of an acquired tacit knowledge into specified instructions to be followed mechanically and transparently. An example of this transfer is an engineer who designs procedures that describe the actions of a sushi master. Once the procedures have been explicitly declared, they can be implemented in a sushi machine.

Combination (Explicit to Explicit) is the transfer of different pieces of explicit data to be compiled and processed into a new piece of data. This is done instructionally. An example of this is a generated report that retrieves information from a number of sources.

Internalization (Explicit to Tacit) is the transfer of knowledge from an explicitly detailed source into such as an instruction manual describing.

Fred Nickols proposed a unified framework [Nickols, 2000] for use in knowledge management, using both Polanyi and Anderson schemes and also including a new category:

Implicit Knowledge: In his framework, Nickols defines this as knowledge that has not, but can be, articulated. The notion of implicit knowledge is unclear in its presentation, but can be conceptualized as a middle ground between pure tacit and pure explicit knowledge.

Nickols’ concept is based upon a flow diagram that links Anderson’s definitions for Declarative/Procedural knowledge with Polanyi’s proposal. The original diagram is found in Fig. 2.

Jeffrey Zheng et al. [Zheng, 2001] constructed a graphically balanced model known as the executable knowledge model, adopting Nickols’ theory and certifying what Implicit Knowledge meant. The idea to this model was that in order for tacit knowledge to be transformed into explicit knowledge and vice versa, it must be first transformed into implicit knowledge. Similar to Nonaka, four transformations were proposed to transfer Tacit and Explicit Knowledge into Implicit Knowledge and back again. The model also suggested that a machine
could implement Explicit and Implicit Knowledge whereas Tacit Knowledge needs to be implemented by a human being shown in Fig. 3.

From the 1990s, T. L. Kunii proposed cellular spatial structures as $n$-dimensional topological spaces ($n$-cell) to use multiple hierarchical structures to resolve knowledge-modeling problems [Kunii, 1998]. Applying recursive constructions using set theory, homotopy [Whitehead, 1949; Baues, 1996], topology and spatial constraints to establish modular and incremental design as an inheritance hierarchy of invariants, this graph-based model has shown its general applicability in many applications to resolve the foundational problems of practical applications. The $n$-cells are successfully applied to synthetic worlds, shape modeling, web information systems and world models [Kunii, 1998, 1999, 1999a, 1999b, 2002].

1.2. Overview of knowledge systems

In 2001, Shin et al. [Shin, 2001] surveyed current theoretical schools and their management practices as four points.

There are three main viewpoints on knowledge management and implementation of knowledge systems:

(a) Mind focuses mainly on the theory that knowledge extends from the mind itself and research concentrates upon creating infrastructure individuals in order for the better understanding of knowledge.

(b) Process focuses mainly on the processes in which knowledge is delivered from a source and recipient. Research concentrates on designing facilities to improve the creation and transmission of knowledge.

(c) Object focuses on the accumulation and manipulation of actual knowledge. Research concentrates on ways to effectively code, store and retrieve wanted knowledge.

(d) All schools of thought agree that knowledge is different from information.

(e) Together all of them identified five main factors that affect the distribution of knowledge: culture, knowledge location, awareness, evaluation and absorption.

(f) Principally, there are two approaches to determine knowledge. One uses the concept of value chain or hierarchical structure among data, information and knowledge. Another one focuses on the analysis of the process of knowing.
2. Network Topology of Knowledge Management

2.1. The arrangement of knowledge

All the models summarized in Sec. 1 have unclear boundaries between tacit and explicit knowledge. Even with the implicit middle tier, there does not seem to exist a clear distinction between when tacit knowledge becomes implicit, and when it becomes explicit. Whilst the models have used similar terms and definitions of knowledge (such as declarative, implicit and explicit), there have been conflicting usages of each term between authors and their models, even though most are derived from the two earliest models.

Since the boundaries of knowledge would be difficult to define, knowledge transfer between the differing types of knowledge would similarly be complex to identify. This paper proposes a multi-tiered system of knowledge organization as a series of acyclic networks. Instead of manually sorting knowledge as a set definition to which each type belongs, a hierarchical organization of knowledge is the elegant solution to provide a mechanism which automatically provides both the type and usage properties of that particular concept. This will be further explained later in the paper.

2.2. An introduction to network theory

The contents of any set of information can be related to each other through the use of flow diagrams. Starting with the simplest flow diagram being the list, whereby each information node links to the next node in series, many other flow diagrams have been produced, including trees and directed graphs. Directed graphs have so far been the most popular model when working with process control, computer architecture, electrical circuits and other such dynamic systems. Examples of these network models are given in Fig. 4.

The acyclic lattice shown in Fig. 4(d) is a special directed graph, where there cannot exist any loops. In fact, an acyclic lattice can model any system which does not contain loops. This type of lattice has a general flow in one direction; when going from node to node, it is impossible to traverse the same node twice. Like all directed graphs, this lattice has four different types of nodes. This is shown in Fig. 5.

Singleton node: A singleton node is an isolated node, without a graphical relation to any other node. There is no flow from a singleton to other nodes.

Source node: A source node has only outgoing flow to other nodes, signaling the beginning of flow. It has at least one directed link to other nodes, but no link from other nodes.

Branch node: A branch node has both incoming and outgoing flows, manipulating the flow of the system.
Sink node: A sink node has only incoming flow from other nodes, signaling the end of flow.

Using this scheme, a set of information or ideas can be sorted into a network of nodes. These nodes would be arranged in either a time-based or hierarchical sequence, depending on the type of knowledge the network represents. It is this network topology scheme which has much to do with knowledge management. Knowledge itself can be arranged into a network of interrelating ideas and concepts, yet in models so far, no emphasis has been placed upon constructing a knowledge network. The acyclic lattice model provides the flexibility to arrange knowledge concepts such that these concepts may be both easily accessible and inter-related. This type of lattice also ensures that the flow of concepts does not infinitely loop, nor is based upon circular logic, both of which could occur in directed graphs.

2.3. Changes to definitions of ontology

Readers using this topological view of knowledge would need a clean slate regarding definitions of knowledge from past knowledge management papers as described in Sec. 1. This paper has attempted to unify all existing knowledge models and conflicting terms in models which so far have been subtly incorporated into a more programmable, yet powerful model.

The first major change the reader is asked to accept is the definition of Declarative and Procedural Knowledge. Declarative Knowledge are definitions of each concept and can be arranged as a static relationship to each other. For example, the terms ‘mother’ and ‘child’ are intrinsically linked, as there can be no child without a mother and vice-versa. The terms have no time-sequenced pattern, hence the relation remains invariant in time. Similar definitions of Declarative Knowledge are linked together within an acyclic lattice arranged from the most abstract to the most defined concepts. Procedural Knowledge uses concepts already listed as Declarative Knowledge, linking them in a time-sequential manner. For example, the concepts of ‘walking’, ‘skipping’ and ‘bouncing a ball’ can be statically related in some way through Declarative Knowledge, but Procedural Knowledge is able to link the three terms together to form another concept — ‘dribbling’ through a time sequence of events.

The second major change would be the re-definition of the terms Implicit, Tacit and Explicit. In most cases, Explicit Knowledge was formerly linked with Static (Declarative) Knowledge, whilst Tacit Knowledge, defined by Nickols as the act of doing, was linked with Dynamic (Procedural) Knowledge. This is not the case for the proposed knowledge model. There are some examples of knowledge being both static and tacit, such as ‘love’ and ‘happiness’, which cannot be dynamically defined nor explicitly described. There are also examples of procedures such as ‘chess program’ or an ‘industrial production line’ where every move can be explicitly defined, yet each is an act of doing. In order to resolve this confusion, Implicit Knowledge is defined as a declarative concept more abstract than Explicit Knowledge. Tacit Knowledge, also a declarative concept, contains even more abstraction.

Using the proposed knowledge model, all concepts can be declared and all must be declared. Each set of concepts is declared in the form of a Declarative Knowledge Lattice (DKL). The DKL defines the static relationship between concepts through an acyclic lattice structure where concepts are linked to each other using a topological network based upon their uncertainty. The lattice would only be concerned with the degrees of relative uncertainty (whether a concept is more uncertain than another), hence the terms explicit, implicit and tacit can be used. The uncertainty for any concept would depend on the situation or the knowledge system, so the abstractness of concepts can only be compared with each other within a DKL, but not between.

The third major change is the introduction of time-based knowledge functionality in the form of a Procedural Knowledge Lattice (PKL). Every method can be defined as a number of small individual actions linked together by a time-sequence. For example, even a slightly
A complicated action such as throwing a ball can be divided into a number of different movements of the body in the following order (grip the ball, bend arm, bend legs, start straightening arm, start straightening legs, release ball in the act of straightening). All these actions can then be separated even further; for instance, to describe the movements required to bend an arm, the individual movement of muscles may be required. Finally, through explicitly defining each individual movement of the arms, the act of throwing a ball can be defined as a series or multiple series of tiny actions occurring in a time-sequential manner. It makes no sense, for example, to release the ball before it has been gripped or to straighten the arms without bending them first. It is essential to establish a sequence of events in which concepts should occur. That is the core function of the PKL.

A DKL is necessary to define a PKL. After all the concepts have been first defined by the DKL, the PKL links them in a time-sequential order. In this way, the two lattices model both static and dynamic relationships of a set of concepts. The functionalities of any concept, therefore, would then be described by two relationship networks to other concepts in the system — one with time invariant (DKL), and one without (PKL).

### 2.4. The Declarative Knowledge Lattice

The Declarative Knowledge Lattice (DKL) represents an invariant structure similar to a traditional data structure hierarchy. In many ways, the DKL is similar to a class type model in terms of its construction. For instance, the DKL has declarations of nodes or concepts arranged in an acyclic lattice structure, where each of the declared concepts is a node in the lattice. The description of knowledge held by a DKL (whether this concept is tacit, explicit, implicit or extensive knowledge) is described from the four types of nodes of network topology. In relation to knowledge management, the determination of the types of knowledge are intrinsic to the lattice itself. Once the lattice is generated and a network is sequenced, the flow of the nodes themselves determines the types of knowledge they represent. A diagram is shown in Fig. 6 to provide a clearer visual understanding of the DKL.

The DKL in Fig. 6 contains nodes labeled a, b, c, d, e, f and g. Figure 6 shows a prearranged lattice in which the seven nodes exist. These nodes represent concepts related to each other through degrees of uncertainty, from 0 to 1. Uncertainty is the measure of certainty. For example, if a concept has a low uncertainty such as 0.1, it is very definite and can be quite clearly conceptualized, whereas a concept with high uncertainty, such as 0.9, is a general description of a concept which may require additional information to conceptualize. The nodes can be defined and arranged as four knowledge categories:

- **Tacit Knowledge** can be defined as the set of source nodes (cells) in the DKL. For any given knowledge base, tacit knowledge has the highest uncertainty.
- **Explicit Knowledge** is defined as the set of sink nodes in the DKL. Explicit Knowledge has the lowest uncertainty.
- **Implicit Knowledge** is defined as the set of branch nodes in the DKL. Degrees of implicit knowledge can also be determined from the relative distance a branch node is from a source, or sink. Its measure of uncertainty would be between the uncertainties given for tacit and explicit knowledge nodes.
- **Extensive Knowledge** is defined as the set of singular nodes in the DKL. This concept of knowledge is introduced to describe a node providing a complete concept, unrelated to any other nodes.
Note that the concepts of the types of knowledge in this model differ greatly from previously proposed models. In this model, Declarative Knowledge encompasses Tacit, Explicit, Implicit and Extensive Knowledge as all of these can be transparently declared at a given moment in time. All of these are static representations of knowledge, related to each other by degrees of uncertainty.

To show how to link with common concepts to Declarative lattice, a short case is selected [Kunii, 1999a]. Interested readers can delve into more elaborated explanation from Kunii’s original paper.

For example, in mathematics, modeling of mathematical objects is conducted to classify mathematical object into equivalence classes as a disjoint union of the subsets of objects by an equivalence relation that represents mathematical invariants. A case of an abstraction hierarchy of equivalence relations as follows:

1. Set theoretical equivalence relations;
2. Extension equivalence relations, homotopy equivalence relations as a special case;
3. Topological equivalence relations, a graph theoretical equivalence relations as a special case;
4. Cellular spatial structure equivalence relations;
5. Information model equivalence relations;
6. View equivalence relations.

In relation to make an incrementally modular inheritance hierarchy of invariants of Cyberworlds, an abstraction hierarchy of invariants can be designed as follows:

1. A set level;
2. An extension level, a homotopy level as a special case;
3. A topological level, a graph theoretical level as a special case;
4. A cellular spatial structure space level;
5. A information model level;
6. A presentation level.

To represent each level of Cyberworlds as a Declarative node, a Declarative lattice can be constructed in Fig. 7.

Applications of the DKL and the use of the lattice can be seen in Sec. 4.

2.5. The Procedural Knowledge Lattice

The Procedural Knowledge Lattice, or PKL, has an analogous structure but different properties to that of the DKL. The PKL relates concepts and their functionalities in process modeling, by arranging them via a time-sequenced lattice of control nodes. The flow of a sequence of concepts is determined by the arrows linking each of the concepts through time. This is shown in Fig. 8.

The PKL in Fig. 7 also contains nodes labeled a, b, c, d, e, f and g. For the seven nodes to exist, they must have already been declared inside a DKL. Once declared, the nodes can be used in a procedure-like manner as a time-sequence of one concept after another. Although a concept can only be statically declared once, a concept can be dynamically used and reused over and over again. Although this type of reuse
of concepts differs from DKL, there are similarities due to the acyclic lattice structure of the two. Like the DKL, nodes in the PKL can be sorted into four differing categories, determining the types of procedural knowledge:

Start is defined as the set of source nodes in the PKL. Start nodes signify the starting concept/s of any procedure — simple or complex.

Action is defined as the set of branch nodes in the PKL. Action nodes signify each individual action or concept required to link the start and end of a whole procedure. Most of the nodes in any PKL would most likely be action nodes.

Finish is defined as the set of sink nodes in the PKL. Finish nodes signify the end concept of the procedure.

Operation is defined as the set of singular nodes in the PKL. Operation nodes represent a complete procedure by itself, encapsulating all of start, action and end nodes in a single operation node.

The PKL would be easier to conceptualize than the DKL. Any given procedure requiring a set amount time to complete can be modeled using the PKL. The difference between the PKL and the DKL can be seen by using the analogy of a computer program. Declarative Knowledge can be seen as the source code of the program. In a object orientated programming, the DKL of the program can be constructed based upon the abstractness of each object in the program. Procedural Knowledge does not exist with programming and examples of the knowledge structure can only be seen when the program is executed. The complementary nature of the DKL and PKL can be seen in an example of a basketball player attempting a shot. Declarative knowledge of the shot would be mechanisms used to control each small individual movement of muscles required to make a shot. Procedural Knowledge would be the shooting action — knowing the order of the movement of muscles whereby a time-sequenced procedure is performed so that the ball is released out of the player’s hands into the ring.

Another significant difference between DKL and PKL nodes are in their type classification. DKL nodes are classified as only one of the four Declarative categories (tacit, implicit, explicit, extensive), PKL nodes however, can be classified in one of more categories of Procedural categories (start, action, finish, operation). This could be represented by the fact that a procedural node is a node in both time and position. Therefore, if node a is an abstract Procedure node, then a(t,p) is its instance in time t and position p. Under this notation, there is no problem to retain any Procedure node in a unique category. Conventionally a simplified representation does not include the input parameters.

From a practical view point, which node needs to be in start or finish category completely depends on situation. For example, in business environment, many business contracts may be agreed by corporators in their headquarter office after hard negotiations. In the case, it is necessary to have agendas, documentations and listed issues to record time events, confirmed items, coming topics and the final goal. Both parties may reach the goal through a series of formal meetings. In alternative, some business contracts may be agreed in different environment such as golf court, swimming pool even in the elevators. To use Procedure lattices to show such significantly alternative actions, the same actors may be involved, and other related concepts contained in one DKL. But which Procedure node must be in the start or finish category strongly relates to the most desired conditions to let actors in the best performance (chat, joke, visual aid, greeting, welcoming, drink, story, question). How could we select a proper Procedure node to put into the start category? There is no absolute answer. The selection may depend on the cultures in society, business traditions, personal styles and competition advantages.

3. Introduction to the Concept Cell Model

3.1. The Concept Cell

The most basic examples of life are cells, the simplest functioning package for any given organism. Life can be contained in one cell (bacteria),
yet cells can also interact with one another to form organisms with extreme complexity (humans). Within an organism, each cell has the capacity to interact with other cells such that both the complete organism and the cell itself can grow. The growth and organization of cells have remarkable similarities to knowledge and its growth. The cells not only need to have control of declared movements and functions, but they also need to co-ordinate these movements over time in order for further development.

The key to knowledge systems may in fact lie in the organization of these basic yet complete building blocks. Using the analogy of life construction to knowledge organization, an artificial cell known as a concept cell is proposed as an abstract model to represent the most primitive of building blocks for knowledge. Packaged similarly to a biological cell, the concept cell comprises of a membrane (interface with external objects, that is, other cells), cytosol (foundation and base content) and a nucleus (enclosing properties for knowledge and growth of knowledge). Just like the biological cell, the concept cell holds all information and functionalities needed to interact with other concept cells, to share information and most importantly, to link individual concepts together to form a complex procedure.

3.2. Concept Cell Overview

Note that a concept cell represents an artificial interpretation of a particular concept, therefore, this paper uses the terms concept and cell interchangeably as both represent the same idea. A typical Concept Cell (K) comprises of three integral components — the nucleus, the cytosol and the membrane, explained in detail below:

**Nucleus (N):** The nucleus is the foundation of the cell whereby the cell’s core functionality is contained and expressed. A nucleus can contain one or two sub-nuclei where lower-level concepts are declared and organized into acyclic lattices:

- **Declarative Nucleus (D):** contains one or more Declarative Knowledge Lattices (DKL). Each DKL first declares then organizes lower-level concepts in terms of their static relationship with one another.

- **Procedural Nucleus (P):** contains one or more Procedural Knowledge Lattices (PKL). Each PKL organizes each of the concepts in the Declarative Nucleus in terms of their time-sequence to form a more complex procedure.

To avoid confusion between the with Procedural Knowledge Lattice (PKL) nodes, and the Declarative Knowledge Lattice (DKL) nodes, DKL nodes are drawn as rectangles whilst PKL nodes are circular. This visual scheme was first introduced in Chapter 2 and will remain constant throughout this paper. Hence, the Declarative Nucleus is represented as rectangular whilst the Procedural Nucleus is represented as an oval.

One difference between the Concept Cell nucleus and the biological cell nucleus is that the former has direct access to the membrane so that the nucleus itself is able to efficiently interface with external objects. The Concept Cell nucleus and it sub-nuclei provide a holder for the two types of Knowledge Lattices. The functionality and complexity of the nucleus is determined by the complexity of the acyclic lattice structures.
A nucleus can be either a Simple Nucleus or a Complex Nucleus:

**Simple Nucleus:** only contains one DKL and one PKL.

**Complex Nucleus:** must contain one or more DKLs. If there’s only one DKL, it must contain multiple PKLs.

Note that the determination of a nucleus type (whether it is simple or complex) only depends on the number of DKLs and PKLs in the nucleus. A cell containing a single intricate Knowledge Lattice Network of many nodes would still be defined as a Simple Cell, whilst conversely, a cell containing two Extensive concepts (singleton nodes) would be defined as Complex as the two concepts represent two networks.

**Cytosol (C):** A base description of the content, containing the global properties and documentation of cell structure and function. Cytosol, in a practical programming aspect, defines the type and default value of the cell.

**Membrane (M):** Containing both the nucleus and cytosol, the membrane provides interfaces for the cell to interact with other Concept Cells.

The Concept Cell is visualized in Fig. 9.

The cell has certain properties of symmetry to its construction based on the construction of the two knowledge network lattices. Both have an absolute construction of a lattice (Extensive Knowledge Nodes for DKL and Operation Nodes for PKL) and both have a general construction of the lattice (the Nodes linked with arrows). It is important to note that the boxes represent a set of knowledge nodes, not just a single one. Another method to visualize the concept cell would be through ontology. This method is shown in Fig. 10.

The internal relationship of all basic structures can be represented in the above Ontology diagram. This hierarchical structure provides the foundation for all concept cells.

### 3.3. Basic Concept Cells

A concept cell is constructed from these three components, and the complexity of the cell depends only on the complexity of the nucleus. A nucleus may contain any number of Declarative and Procedural Knowledge lattices but usually, a nucleus will not have more than one of each type of lattice. This is known as a simple nucleus and, when combined with a membrane and cytosol, forms a simple cell. There are three categories of the most basic cells and each are shown in Fig. 11.

Base Cells are the most fundamental of all Concept Cells, having an empty nucleus. The Cell only has cytosol properties and, used in a programming aspect, can be used to define the

![Fig. 9. Modeling of the Concept Cell.](image)
Simple Cells have a nucleus containing both a DKL and a PKL. This is used to define concepts which can only be expressed through a time-sequence of events. Knowledge is modeled using a simple cell. Depending upon the complexity of lower-level cells, or the complexity of all the lattices, Simple Cells can be either very simple or very complex. Examples of data types include ‘Chess Game’, ‘Respiration’, ‘Dying’ and when expanded to become a complex nucleus, can model any concept.

Each of these cells contribute in different ways to the final model. In general, the lowest level cells (meaning the most basic components that need to be implemented) are the Base Cells containing the definition of basic element types. Simple Declarative Cells may utilize Base Cells, relating the concepts that a particular cell represents in a DKL. Each of the concept cells are linked with other cells, starting from tacit (most uncertain) knowledge flowing down to explicit (most certain) knowledge.

In order for the system to model a concept involving a time-sequence of lower-level concepts, a PKL is needed. The PKL allows for a process to be modelled. Walking, for example, is a process requiring the co-ordination of a series of movements of muscles in our body. We can all grasp the concept of walking but a computer cannot. ‘Walking’, the concept, not only requires the definition more basic movements, it requires them to be synchronized into a process. Therefore, only using the DKL is not...
enough and a PKL is needed. Examples such as walking and other higher level knowledge models would require at least a Simple Cell in order to be defined. Hence, concepts involving multiple processes would require multiple PKLs and Complex Concept Cells.

3.4. Complexity of Basic Cell types

If all the nodes of the lattice were to be viewed as sets of nodes with similar properties (i.e. singleton nodes, source nodes, branch nodes and sink nodes), using the four generic node sets, any acyclic lattice network can be simplified to form three configurations:

- Single-Tier Structure: Singleton
- Two-Tier Structure: Source → Sink
- Multi-Tier Structure: Source → Branch → Sink

As both the Declarative and Procedural Knowledge Lattices are acyclic lattices, they would each be able to form three configurations. The configurations of each of the lattices can be seen in Fig. 12. Therefore, both knowledge lattices can each assume three forms and the construction of even basic cells have great scope for variations.

Table 1 shows the combinations of knowledge cells that occur with combinations of all the basic elements.

Considering that the variations of Base Cell are formed entirely from the two knowledge lattices (D and P), the number of combinations (13) available for the construction of a basic cell is very high, thus giving such a simple system a great deal of flexibility to handle the simple knowledge systems (i.e. systems that do not need a lot of definitions of its variables). Knowledge handling and management requiring a number of definitions could still work with the concept, but an effective method built around the layering of concepts is introduced.

3.5. Concept Cells layering

When the two types of Knowledge Lattices were introduced in Sec. 2 and further developed in this section, the content of each Knowledge Lattice was referred to as a ‘node’.

Table 1. Comparisons of cellular model.

<table>
<thead>
<tr>
<th>Cell Name</th>
<th>Components</th>
<th>No. Forms</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base Cell</td>
<td>Membrane (M), Cytosol (C)</td>
<td>1</td>
<td>The simplest element. No Knowledge Lattices and therefore only has one form.</td>
</tr>
<tr>
<td>Simple Declarative Cell</td>
<td>M, C, Nucleus (N), DKL (D)</td>
<td>3</td>
<td>Base Cell and a DKL. Can assume three forms due to the Declarative Lattice.</td>
</tr>
<tr>
<td>Simple Cell</td>
<td>M, C, N, D, PKL (P)</td>
<td>9</td>
<td>Base Cell with one each of DKL and PKL. Because both can assume three forms, the total combination would be $3^2$ or 9 forms.</td>
</tr>
<tr>
<td>Total</td>
<td>M, C, N, D, P</td>
<td>13</td>
<td>The combined total of the three different cells.</td>
</tr>
</tbody>
</table>
Main Cell

Lower Level Cells

Each ‘node’ was regarded as a complete concept and this was first declared and ordered in a DKL and then implemented in a PKL. With the introduction of the Concept Cell model, it is clear what the ‘node’ explicitly represents; each node is a Concept Cell. This can be seen more clearly in Fig. 13.

Figure 13 shows how a concept can be generated using smaller concepts. The main cell holds a DKL and a PKL with nodes \{a, b, c, d\}. These nodes are in fact smaller concept cells, each with their own Declarative and Procedural Nuclei, containing lattices of simpler concept nodes.

Using the concept cell representation, any level of knowledge, no matter how abstract nor basic, can be represented and linked using levels upon levels of cells, each containing definitions of lower-level concept cells required to make up the contents of the cell. Conversely, the cell would also be combining with other cells to form knowledge lattices in order to construct a higher complexity of knowledge. In this way, any concept can be either constructed through a bottom-up process from definitions of the most basic data types, or top to bottom by deconstructing a difficult concept to give smaller concept cells for further deconstruction. Construction and deconstruction of concepts could be both very simple or very complex (from one or two layers to a hundred) and this would depend on the application for which the concept cell model represents and the detail required by the user.

4. Concept Cell Organisation and Networks

4.1. Extending the Simple Cell: Restaurant service examples

Simple Concept cells can be extended to become a Complex Cell if more lattices were added into the cell. This would arise if there were more than one method of relating concepts such as when a concept has more than one meaning or usage (i.e. wind could be used in two different contexts — ‘A gust of wind’ or ‘Wind up the clock’) by adding more DKLs to the Cell. More commonly, a simple cell can be extended when there are multiple processes that would end in the same result. This is done through adding PKLs to the Cell. To better understand this process, a knowledge model of services in different types of restaurants is constructed.

In a restaurant, the most explicit concepts are that of food and money. Money is exchanged for food using four concepts that we are familiar with — order, pay, serve, tip. The agents with which all these concepts interact would be the customer and the worker. These are the basic concepts needed for a simple restaurant model. We will represent the eight concepts as base cells for simplicity’s sake. It is evident that all of these concepts can be further broken down into simpler concepts — food for example, can be broken down into simpler concepts like meat and vegetables and they can be broken down further and so on — yet in this model, it would be enough to view the eight concepts as the ‘base’ concepts. How much definition is supplied to any concept is entirely the discretion of the user.

To construct a DKL, all of the eight node are defined and arranged in a lattice with the most tacit of concepts (customer and worker) being the source nodes. They have been chosen to be the most tacit due to the fact the terms can represent anyone representative of the restaurant or anyone being serviced by the restaurant. The terms order, pay, serve and tip are all concepts of the communication and trade processes and the connotation that each concept represent...
would be less uncertain than the two most tacit
nodes. The explicit nodes (food and money) are
the most certain of concepts and the only uncer-
tainty would be the type of food coming off
the menu or the value and denomination of the
money paid. The DKL is constructed in Fig. 14.

The most important objective in establish-
ing the DKL relationship is to be able to estab-
lish links between the tacit, implicit and explicit
nodes. In this way, the serve concept is intrinsi-
cally linked with worker as can be seen from the
DKL, whilst money is intrinsically linked with
both pay and tip. Therefore, with the static rela-
tionship established, the PKL would be easily
defined.

With the eight basic concepts defined, the
procedures involved with service in a restaurant
can then be defined. Depending on the type of
restaurant, the procedures work differently. For
example, in a fast food restaurant, customers
first order, then pay, then are served by the
worker. However, in a Normal Restaurant, the
Customer orders and are served by the worker
before paying. In many instances, workers would
also require tips for their service. Another exam-
ple of a variation of procedure would be when
going to Yum-Cha. The customer would order
and be served, order again and be served and
this process would continue indefinitely until
the customer is full, finally the whole meal is
paid for. Each of these processes would require
a separate PKL to model as they are separate
processes, each relating to the same concept
of servicing in a restaurant. Figure 15 shows
all the PKLs constructed using the eight base
nodes. The PKLs are labeled \{P1, P2, P3\} for
convenience.

The complete Complex concept Cell model-
ing this process is shown in Fig. 16. The PKLs
P1, P2 and P3 represent servicing for a Fast-
Food Restaurant, a Fine Restaurant and a Yum
Cha Restaurant respectively.

More Procedural Knowledge Lattices could
be added if needed, as long as the concepts in the
PKLs have been declared. For example, if the
Restaurant Servicing Concept needed an exten-
sion of a Buffet Service, more concepts would
have to be defined such as the self-servicing of
food. Once all the concepts have been defined,
they would be able to be applied within a Buffet
Service PKL.

---

**Fig. 14.** Ontology of the Cell Model.

**Fig. 15.** PKLs of Restaurant Servicing.
4.2. Further extensions: A shopping complex example

The Concept of Restaurant Servicing can be mapped out to a concept cell. Using this concept, we will attempt to extend the knowledge base by modeling a Shopping Complex and the services offered. Services would include Clothes, Toys, Cars, Electronics and Food Services. Using the above model, we can model a shopping complex that have multiple centers servicing the same goods. For the moment, only Food Services would be taken in depth.

Consider a Shopping Complex with one each of Clothes, Toys, Cars, and Electronics Service but five restaurants. Two are Fast Food Restaurants (KFC and McDonalds), two are Fine Restaurants (Greek and Italian) and one serving Yum Cha (Chinese). All the restaurants have the servicing model of customers based on the Concept Cell model of restaurant servicing proposed in Sec. 4.1.

As the basic function of a Shopping Complex is the conversion of money into services, it would operate the same way as the restaurant model. The declarative lattice of a Shopping Complex Concept Cell would consist of:

**Tacit Nodes** \{customer, worker\}

**Explicit Nodes** \{money, food, toys, clothes, cars and electronic goods\}

**Implicit Nodes** \{set of all the services list above, examples being Clothes Store, Garage, KFC, McDonalds, Greek Restaurant etc...\}

The DKL of a Shopping Complex is drawn in Fig. 17.

Figure 17 shows the static relationships of all involved nodes. All the services that are not food are condensed into a single set of other Services. Each node in that set would require a Concept Cell similar to that of the Restaurant Concept Cell. Since the Restaurants are only linked with exchanging food for money via customers and workers, the DKL would not link the restaurants to any of the other explicit nodes such as clothes, toys or cars. Note that the five restaurants are shaded in a different color to all the other nodes as to emphasis that they are all cells of the Restaurant Servicing Concept proposed earlier. The labels are provided so the reader can distinguish one from the other.

Each of the five restaurants would need systematic methods of distinguishing their business against each other. This would be where the Cytosol is needed for all the basic information of the Restaurant. Although all the information of the Restaurant Servicing Concept is contained in all of the 5 instances, not all the concepts are expressed. This is done via the additional information contained in the Cytosol. Diagrams of concepts modeling the KFC restaurant and the Chinese Restaurant (Shark Fin) can be seen
in Fig. 18. Note that additional information could be contained in the base about a variety of properties, yet only a few are listed.

Note that the other three restaurants would be constructed in this fashion, inheriting from the constructed Restaurant Servicing cell and then adding additional information through the Cytosol. The DKL labeled D and the PKLs labeled P1, P2 and P3 have been predefined in earlier sections.

Once the Declarative Knowledge Lattice has been established, the Procedural Knowledge Lattices can also be drawn. This would be relatively easy. Since a Service can already be defined as a procedure, the procedure in already contained within the concept cell itself. Therefore, when a customer is serviced either a restaurant or any other store, the procedure itself would be predefined as a set procedure.

With the shopping complex, a huge number of procedural lattices can be established due to the nature of the number of services on offer. For example, if a customer was to only eat at one single restaurant or shop at one single store, the PKL would be shown in Fig. 19.

The service could be KFC for example.

If the customer wanted to service his car and went to the Shopping Complex in the morning to drop it off. Then because he wanted to occupy himself whilst the car was in service, he shopped for a stereo at the Electrical Goods Store and then bought some new clothes, got hungry and went to McDonalds, the PKL would be drawn as in Fig. 20.

Note that the Definitions in the PKL must be declared in order to be used by the concept cell. A Garage must offer a Drop-Off and a Pickup Service as if this is not the case, the customer would have to wait until the servicing procedure is completed before buying the stereo, clothes and food. Therefore, a Garage Service concept cell would have to be slightly different to that of the Restaurant concept and offer servicing whilst the customer is away.

As can be seen by the large number of combinations of set procedures able to be completed by the customer whilst using the Shopping Complex model, a highly complex Concept Cell would have to be designed. Yet, even if PKLs are many, the static relationship between all of the DKL concepts hold true and so a non-varying property of the cell would always be
Fig. 21. Concept Cell of a Shopping Complex.

there. The Complete Cell of the Shopping Complex would simplify to a Cell shown in Fig. 21.

The Shopping Complex Cell was constructed with DKL DD and PKLs PP1 to PPm. This becomes a complete component, ready to be utilized in a bigger system.

The relative properties of the cell model compared with other schemes are shown in Table 2. In the table, TM represents the Theoretical Model that is used in Knowledge System applications. ST denotes Structural Theory that uses structured organizations to represent complex dependency among members. ES indicates Engineering Systems that provide mixed theories, experiences and skills with commercial system modeling tools for pragmatic applications.

All of these models are crucial to diverse and integral applications in all practices requiring management of knowledge. Such examples include the following industries: Enterprise Management, Manufacturing and Building Industries, Software and Hardware Systems, Global Communication Networks, World Wide Web and Internet Environments. The figure compares the Concept Cell Model to all of the schemes that have been developed so far.

From this comparison, it is clear that the existing systems most similar to the concept cell model come from Enterprise Modeling. Enterprise Modeling provides all functionality for ten meta-nodes from engineering practices. However, most of the other theoretical models cannot provide full theoretical support of those

<table>
<thead>
<tr>
<th>Model</th>
<th>D</th>
<th>T</th>
<th>I</th>
<th>E</th>
<th>X</th>
<th>P</th>
<th>S</th>
<th>A</th>
<th>F</th>
<th>O</th>
</tr>
</thead>
<tbody>
<tr>
<td>Concept Cell</td>
<td>ST</td>
<td>ST</td>
<td>ST</td>
<td>ST</td>
<td>ST</td>
<td>ST</td>
<td>ST</td>
<td>ST</td>
<td>ST</td>
<td>ST</td>
</tr>
<tr>
<td>Polanyi (1943)</td>
<td></td>
<td>TM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Anderson (1973)</td>
<td>TM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Nonaka (1991)</td>
<td></td>
<td>ST</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Nickols (2000)</td>
<td>TM</td>
<td>TM</td>
<td>TM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Zheng et al. (2001)</td>
<td></td>
<td></td>
<td></td>
<td>ST</td>
<td>ST</td>
<td>ST</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lattice Theory (Birkhoff 1967)</td>
<td>TM</td>
<td>TM</td>
<td>TM</td>
<td>TM</td>
<td>TM</td>
<td>TM</td>
<td>TM</td>
<td>TM</td>
<td>TM</td>
<td>TM</td>
</tr>
<tr>
<td>Ontology Metalogic (IDEF) n-Cell (Kunii)</td>
<td>TM</td>
<td>TM</td>
<td>TM</td>
<td>TM</td>
<td>TM</td>
<td>TM</td>
<td>TM</td>
<td>TM</td>
<td>TM</td>
<td>TM</td>
</tr>
<tr>
<td>Conceptual graphs/First order logic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(Sowa 2000)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Enterprise modeling</td>
<td>ES</td>
<td>ES</td>
<td>ES</td>
<td>ES</td>
<td>ES</td>
<td>ES</td>
<td>ES</td>
<td>ES</td>
<td>ES</td>
<td>ES</td>
</tr>
<tr>
<td>Data &amp; Ontology (IDEF0-1-2-5)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Process &amp; Function (IDEF0-3)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Object Oriented</td>
<td>ES</td>
<td>ES</td>
<td>ES</td>
<td>ES</td>
<td>ES</td>
<td>ES</td>
<td>ES</td>
<td>ES</td>
<td>ES</td>
<td>ES</td>
</tr>
<tr>
<td>Function/Procedure/Structure</td>
<td>ES</td>
<td>ES</td>
<td>ES</td>
<td>ES</td>
<td>ES</td>
<td>ES</td>
<td>ES</td>
<td>ES</td>
<td>ES</td>
<td>ES</td>
</tr>
</tbody>
</table>

Base Symbols:
Ten Meta Nodes: D: declarative; T: tacit; I: implicit; E: explicit; X: extensive; P: procedural; S: start; A: action; F: finish; O: operation.

ten types of nodes. This property indicates the potential capacity when the concept cell model is applied to practical applications needing full theoretical support. The Concept Cell Model, when used with Enterprise Modeling, could be of great value to the progress for the development of real-world applications relating to knowledge management and ontology.

5. Conclusion

This paper proposes a hierarchically organized and structured model to represent knowledge consistently. As demonstrated through examples of the restaurant servicing and the shopping complex extensions, the use of Concept Cells provides a strong hierarchical construction to support growth from a simple concept to a complex with layering of concepts.

Construction of a consistent model is only the first step towards practical applications. The authors would like to see further developments to use this model to resolve the practical problems and difficulties in real world knowledge management applications.

It is necessary to develop this model in both theoretical extensions and practical applications. Further engineering approach will help develop this model into more powerful design tools. The Concept Cell approach incorporates and also extends other previous models such as object-oriented, component-oriented and enterprise modeling methodology. It is hoped that future applications would be modelled using this powerful tool. Whilst providing a solid theoretical foundation, the model is still only theoretical and further refinement of this model for practical applications would result in superior knowledge system engineering toolkits resolving real world problems.

Acknowledgments

Thanks to Ms. W. Macmillan, Dr. A. Tharumarajah, Dr. J. Mo, Dr. M. Zhou, Dr. L. Nemes, Dr. G. Liu and Dr. W. Wen for their invaluable comments, suggestions and careful corrections of the early versions. Thanks go to the reviewers in providing invaluable comments in refining the final version of the paper. Thanks also go to Natasha Sung for her invaluable contributions in the proofreading and error checks in the paper.

References

IDEF Family of Methods — A Structured Approach to Enterprise Modeling and Analysis (IDEF0 ~ 5). http://www.idef.com


