A Meeting of the Minds:

A Successful Systems Engineering Experiment using Concept Maps for Effective Communications

John L. BeVier
John L. BeVier & Associates, LLC
1350 Governor Bridge Road
Davidsonville, MD
21035
410-798-4055
mailto:jbevier@erols.com

Colleen A. Calimer
Boeing, Inc.
6919 Beech Avenue
Baltimore, MD
21206
410-663-9846
CPalmer67@comcast.net

Abstract: An informal experiment was conducted to test the effectiveness of communicating Systems Engineering concepts with customers using Concept Maps. The goal was to understand systems engineering concepts and techniques with customers using non-technical diagrams, as well as to establish that the systems engineers had more-accurately captured customer needs.

In short, the experiment was about creating a more customer-friendly vehicle that enabled dialog between the customer and the engineer, without asking the customer to comprehend "Systems Engineering-speak". This paper describes the customer/engineer communication problems found during the experiment, compares the effectiveness of Concept Maps and Use Cases for this experiment, describes Concept Map basics and the customer's reaction to them, and describes the proposed Systems Engineer's use of Concept Maps. The authors note that there are other, successful systems engineering communication methodologies and endeavors. They were not addressed in this experiment.

Never-the-less, the result of the experiment was an enthusiastic and successful meeting of the minds, resulting in a successful shared customer/engineering vision. We offer our Concept Map findings for serious consideration in establishing both customer-friendly and Systems Engineer effective communications.

Introduction: "What we Have here is a Failure to Communicate" Cool Hand Luke, 1967

Everyone knows that the customer is the driver of requirements (he is paying). However, in the systems engineering world, the engineer's emphasis is on the system and as such requirements and system functions are presented to the customer from the systems point-of-view and in the language of the engineering disciplines At the same time, the customer's emphasis is on the mission outcome, not on the tasking details to accomplish that outcome. Even though the engineer wants to understand the desired customer outcome, he queries the customer in terms of the system (i.e.: doing the tasks.) They should focus on the desired system objectives rather than the system tasks. So the basis for the failure to communicate is because the customer and the engineer have not come to a meeting of their minds.

Some attempt is being made to improve communications in the form of Use Cases but still communications problems persist. Systems Engineers decry the failure rate of engineering projects. The customer's allege that there is a communications disconnect between them and the engineers. Additionally, the Systems Engineer seems unaware of meaningful learning concepts whereby the introduction of concepts to the customer should have a definite time sequence because, to the customer, the systems engineering language is not intuitive, and there is a major need for a quick way to convey concepts. It is not for lack of documentation that miss-communication occurs; in fact more documentation is created in an effort to make things clear. There is a better way. It is called Concept Mapping.

Concept Mapping is a language that is relatively easy to understand by both the analyst and the engineer, yet, with enough discipline to be of use to the engineering process. A language is needed that is born from the problem being described so that critical concepts are discussed by the Systems Engineers and by the customer and are accurately and completely understood. The customer is the expert and the systems engineer is the student. So between the two, they must fashion their own language to describe the germane and critical aspects of both of their domains. In effect, they become a learning organization, a team. Concept Maps (Cmaps) can use the most appropriate language to create, display, study, discuss and refine what the customer's needs really are and what the systems engineer will supply that will meet those needs. Afterwards, they can both test the final delivery with confidence against what was agreed upon in the Cmap. Concept Maps, however, are a deceptively simple and elegant solution to the customer / engineer requirements breach. Peers attain the elegance of this solution simply by establishing mutual understanding of the real problem.

"Grant That I Might Not Seek to Be Understood, as To Understand" St. Francis of Assisi

There are multiple reasons for miss-communication by the customer as well as by the Systems Engineer:

1. The customer is busy with real-time mission and operational needs and doesn't have time to devote to teaching the mission, learning the TO BE system and understanding systems engineering.

2. The customer isn't versed in systems engineering and expects to provide information as a one-time push to the developers.

3. The customer isn't really interested in learning and using systems engineering "lingo" and doesn't want to understand how it may help them with their needs. It seems to be unnecessarily detailed and tiresome. Additionally, the Systems Engineering flow of information to the worker level is a new phenomenon; historically, new systems "just showed up."

4. The Systems Engineer/developer thinks that Use Cases are adequate to convey system concepts and customer requirements. It is not recognized that there are further distinctions that are critical to successful application of techniques to the problem at hand or to recognizing the type of problem being attended.

5. The Systems Engineer/developer doesn't think it is their job to fully understand the customer's domain – Customer Subject Matter Experts (SME) have that responsibility and are there for clarification, as/if needed (of course, without the understanding one is unaware that the Customer Subject Matter Expert should be consulted.). Furthermore, there is often a shortage of SMEs.

Introducing New Concepts:

Missing from the Systems Engineering doctrines is the awareness that new ideas must be introduced in a specific sequence, under specific conditions, as researched by David Ausubel (more details provided in the next paragraph). Systems Engineers talk to customers about how they will satisfy requirements and they speak in Systems Engineering language (i.e., Use Cases. The Use Cases describe the TO BE concepts as conceived by the engineer, but those concepts are totally new to the customer. Customers have an inherent fear of the unknown, which is what engineers represent and present to the customer in proposed system solutions. These unknowns multiply to create a threatening background of un-posed and unanswered questions thereby creating additional communication challenges. The engineer also represents change to the existing well known system, which, if altered, may yield results never encountered before including "unintended consequences". Customers bring their own anxiety, suspicion, and a negative attitude to the discussion because engineering is not their domain of expertise and they are forced to contend with new ideas.

According to the learning psychology of David Ausubel (1963, 1968, 1978), meaningful learning of new knowledge is dependent on what is already known. Therefore, Systems Engineers must present their visions for satisfying customer needs in a way that builds upon that which the customer is already aware and in a manner such that it becomes a joint vision. Enter Concept Maps, developed by Prof. Joseph D. Novack of Cornell University (1983), which is a graphic representation of a subject and shows how it is linked to related topics and subtopics.

The Systems Engineer Unwittingly Exacerbates the Communications Problem:

Systems Engineers often embrace Use Case diagrams as a logical way to express processes, what the system will do, and also to capture the scope of the system to be designed/implemented. When Use Cases are shared with the customer, the customer sees himself or herself as an “actor”, and sees a specific part (an instance) of the customer process expressed in terms of “include” or “extend” as it relates to system interactions.

The customer in this experiment had already had disappointing engagements with engineering efforts. Previously, Use Cases were introduced to the customers with less-than-successful results. Customers found the Use Cases to be frustrating because they were not readily understandable; they were described as a “spaghetti” map. The Systems Engineering-speak was limited to two passive verbs, “includes” and “extends”, with no customer meaning. The symbols (bubbles) were mentally regarded as the same; Use Cases were seen as non-intuitive and required explanations. Customers denounced Use Cases as boring and the decompositions were just more of the same. After a few sessions, customers were reluctant to participate and had to be coached in the importance of validating that the system to be built would satisfy customer objectives. Additionally, developing Use Cases was time consuming and use of the Popkin tool requires basic systems engineering knowledge. Time was spent getting the customer to understand Use Case language and in effect, the Systems Engineer brought the customer into the Systems Engineering domain so that the customer could understand the *Systems Engineer’s needs*. The role of the Systems Engineer should be to enter and understand the customer’s domain.

Figure 1 depicts a Use Case example. Later, it will be compared against a Concept Map depicting the same information, in a different “language”.

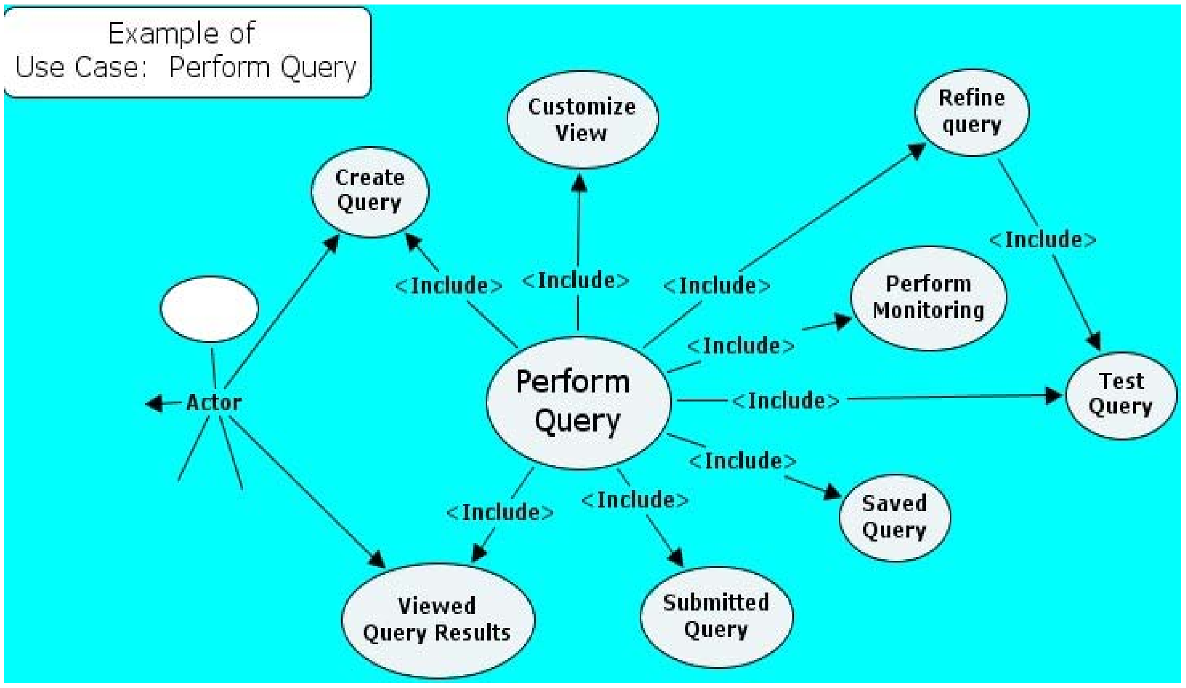

Figure 1: Use Case Example **John L. BeVier and Colleen A. Calimer 2004**

The Experiment: Concept Maps for Cross-Domain Communication

When one author, Colleen Calimer, attended a two-day Concept Map class, she decided to use CMaps in an upcoming Initial Operational Assessment in lieu of Use Cases to convey systems engineering deliverables to the customer. Being aware of the previous communication problems with Use Cases, the decision to use CMaps was quickly embraced by the customer's management. CMaps are intuitive to the customer because they are created in the language that the customer speaks naturally, and not in Systems Engineer-speak. In other words, *the Systems Engineer lives with the customer in the customer's language domain*, capturing and ultimately understanding customer needs. Additionally, CMap becomes the common language and enables dialogue between the Systems Engineers and the customer. Communication is no longer one way - Systems Engineer to Customer- but flows both ways. In the experiment, CMaps were used to present vendor deliverables to independent testers, which will be discussed later.

Concept Map of Use Case:

Figure 2, below depicts the same Use Case as in **Figure 1**, only as a Concept Map. A Cmap is a visual structure for organizing elements that enables understanding. Not only does the information flow both ways, but also both the Systems Engineer and the Customer can grasp the complexity of a situation. In other words, they can both see and understand how bad the situation is, or how much work needs to be done in order to deliver this one "simple" thing asked by the customer. It is a great equalizer, a common denominator. Everyone is on the same page.

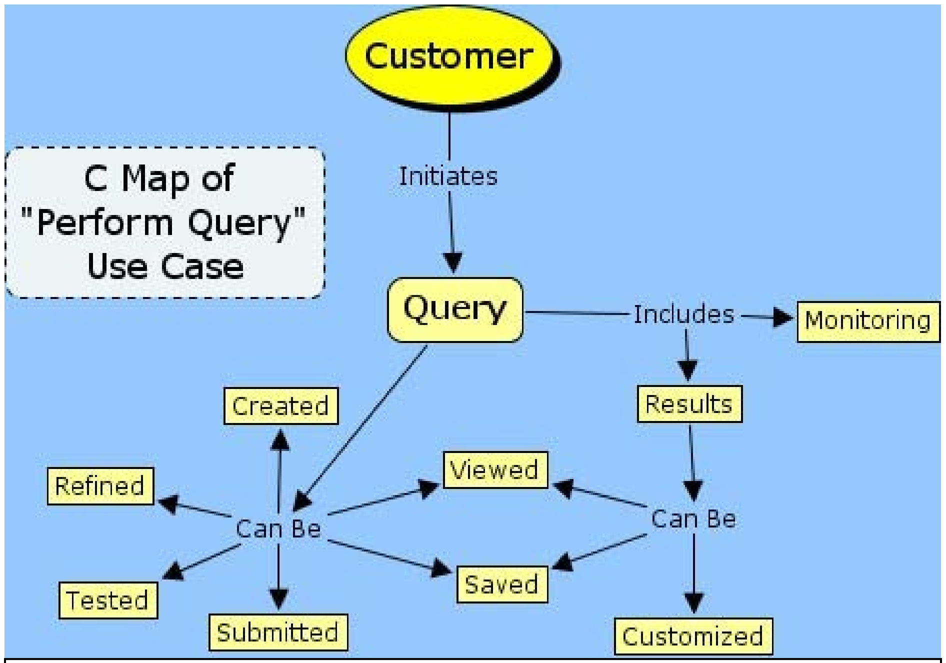

Figure 2: Concept Map of Perform Query Use Case
John L. BeVier and Colleen A. Calimer 2004

As you can see, your eyes travel along the clusters which are logical groupings of concepts; a browse-able knowledge model. The result is an intuitive, immediate comprehension of artifacts and their relationships to one another. An instant Mental Model domain is created in which customer and engineer can reside. No longer is the customer presented with instances of engineering details, without context.

Additional clarity can be added with embedded attachments in the form of documents, web pages, pictures, mouse-overs with comments or explanations, or with other Cmaps. One top level Cmap can express great complexity in this manner.

A Cmap enables customer buy-in from active participation and the results can be shown immediately because the Cmap is user friendly, in the user's language, and easy to modify. Brainstorming and subject mater elicitation is efficiently captured. Creativity is faster since concepts can be "parked" to be filled in later as you build the model.

A brief description of Concept Maps:

"Concept maps are tools for organizing and representing knowledge. They include concepts, usually enclosed in circles or boxes, and relationships between concepts or propositions, indicated by a connecting line between the two concepts. Words on the line specify the relationship between the two concepts. Concept is defined as a perceived regularity in events or objects, or records of events or objects, designated by a label. The label for most concepts is a word. Propositions are statements about some object or event in the universe, either naturally occurring or constructed. Propositions contain two or more concepts connected with other words to form a meaningful statement. Sometimes these are called semantic units, or units of meaning. See **Figure 3** as an example of a concept map that describes the structure of concept maps and illustrates the above characteristics." (Novack)

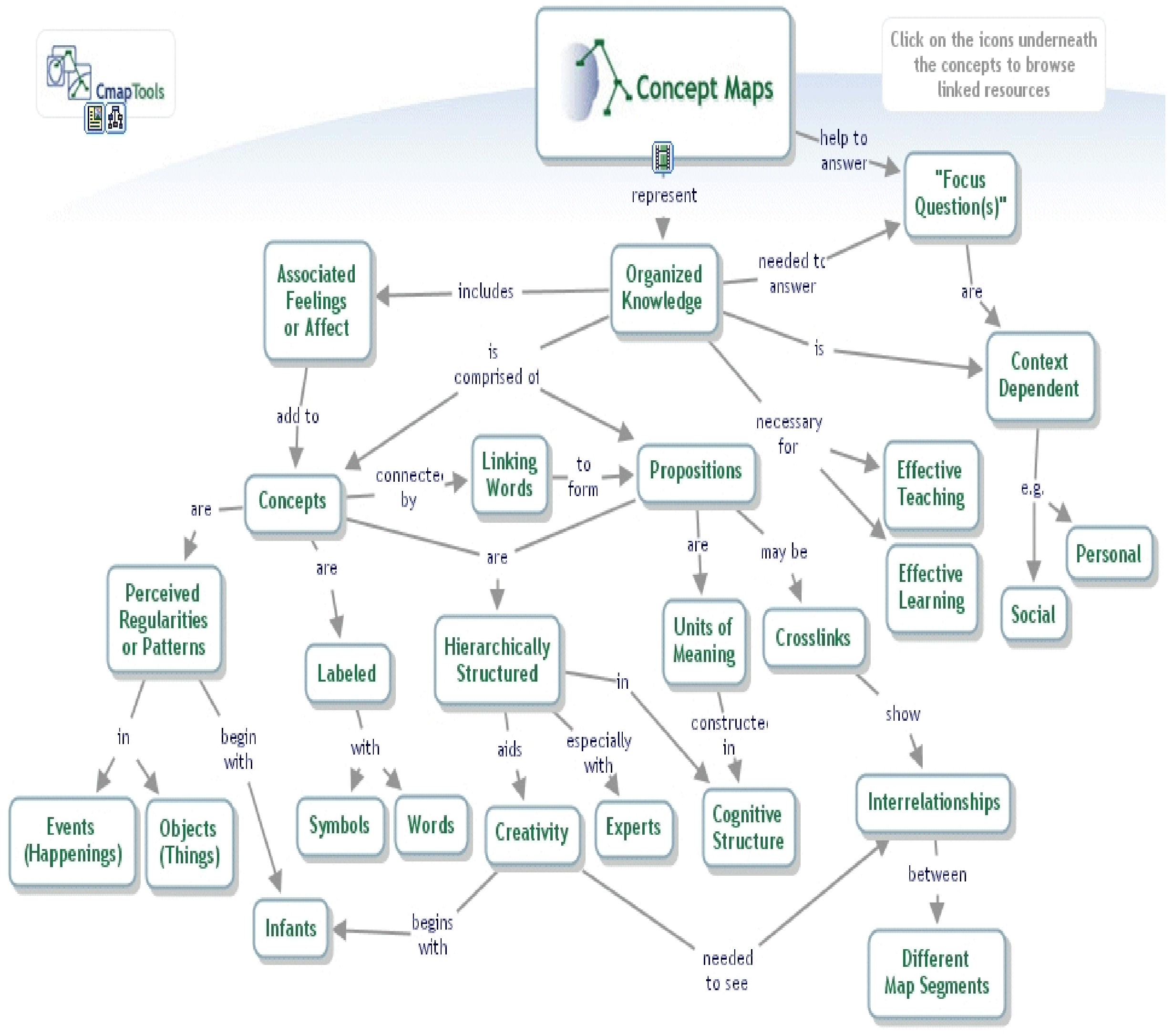

Figure 3: Concept Map of Concept Map Joseph Novack

Now we turn our attention to the visual differences between Use Cases and Cmaps:

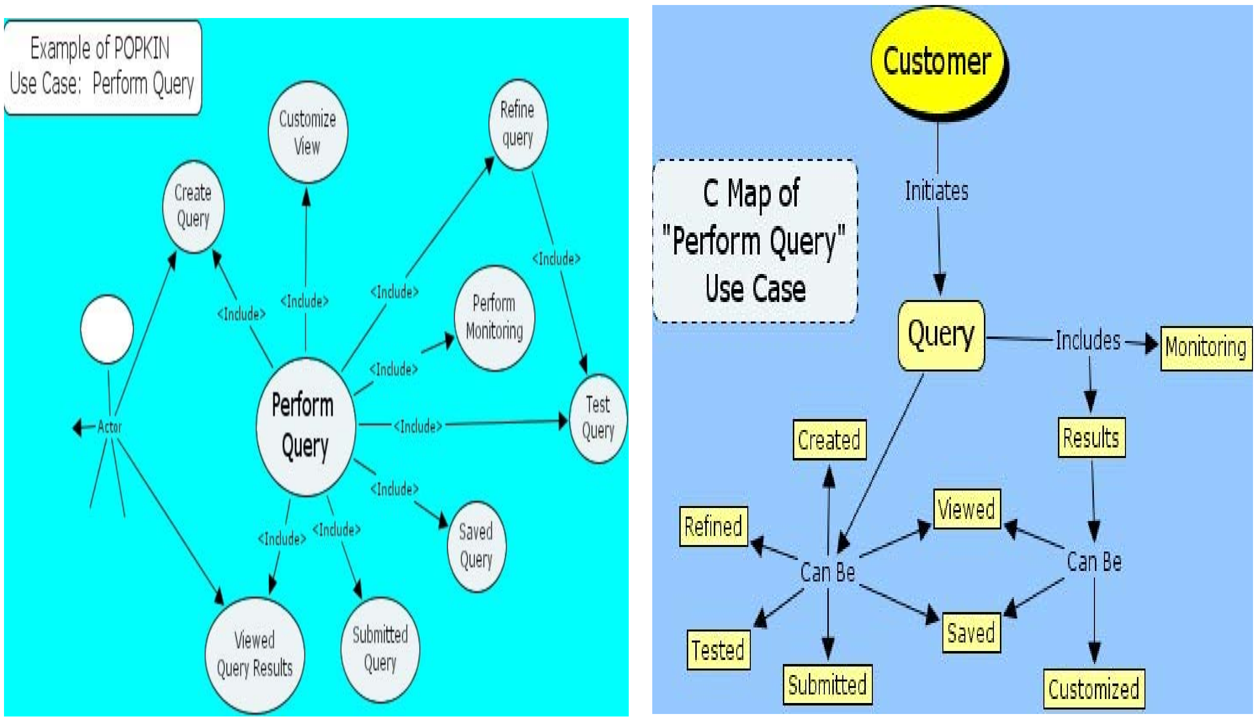

Figure 4 Visual Differences between Use Cases and CMaps John L. BeVier and Colleen A. Calimer 2004

In **Figure 4**, the differences are more obvious when the Use Case and the Cmap are side-by-side. **Figures 1, 2 and 3** were presented to a customer and the reaction was emotional. "I can understand THAT one!" referring to the Cmap version. Although both diagrams present the same information, the desire to understand and to be understood is intuitively and instantly realized in the Cmap version.

Figure 5 compares Use Cases with Cmaps *in the context of communicating to customers, and in the context of the experiment.* <u>Please note that the authors realize that Use Cases are a valuable and necessary tool for Systems Engineering</u>. The Use Cases can define the predictable behavior of a system to the engineers and the text is even more important than the diagram. Use Case diagrams show the overall relationship between the processes and the actors but not all the details.

Visual Differences POPKIN Use Case Cmap “Use Case”	
Comparison in the Context of Customer Communication	
Use Case Diagram	C-Map
Not readily understood = frustration	Visual structure for organizing elements that enables understanding – including how complicated the system and the user’s needs are
One “spaghetti” map	Clustered concepts
Systems Language – limited to two passive verbs with no customer meaning: 1) Includes 2) Extends	Any vocabulary is permissible Unlimited choices Reflects mental model of customer
Symbols are mentally regarded as the same	Symbols, shapes, shadows, colors - all aids in clustering concepts
Decomposition displays more of the same	Attachments to Concepts can be in any form: Documents, Web Pages, Pictures, mouse-overs with comments/ Explanations, other C-Maps, etc.
Non-intuitive; requires explanation	Intuitive; immediate comprehension
Cannot be used for training	Just-in-Time Training
Benchmarks for Operational Assessment by System Engineers	Benchmarks for Operational Assessment by non-Systems Engineers
Developing Use Cases are time consuming	Creation time is faster; concepts can be captured and “parked” for later fill-in
Use Cases require basic systems engineering knowledge. Time is spent getting the customer to understand Use Case language. You must bring the customer to your domain so that s/he can understand your SE needs.	You live with the customer in the customer’s language domain. You are there to understand and capture the customer’s needs.
Customers are reluctant to participate	Customer buy-in from active participation and immediate results
Step-by-Step, linear thinking	Enables brainstorming and Subject Matter Expert elicitation
Diagrams are boring	Immediate Mental models
Post-requirements documentation	Analysts Requirements Gathering
No prioritization; just state diagrams	Requirements prioritization via visual knowledge of what is affected

Figure 5: Use Case and CMap Comparison Chart **John L. BeVier and Colleen A. Calimer 2004**

The Results:

Concept Maps Significantly Improve Communications and Enable More Precise Systems Engineering:

Figure 6 is a fabricated version of a vendor-deliverable for the query portion of a system. Note that in the Cmap, one can mix logical elements with activity elements. This feature creates the context in which the customers' needs are based. The customers see themselves and they see that the system will support their needs.

As you can see, any vocabulary is permissible and the choices are unlimited. Whatever words convey the message so that all understand are the right words. The Cmap reflects the mental model of the customers as well as the TO BE system envisioned by the Systems Engineers.

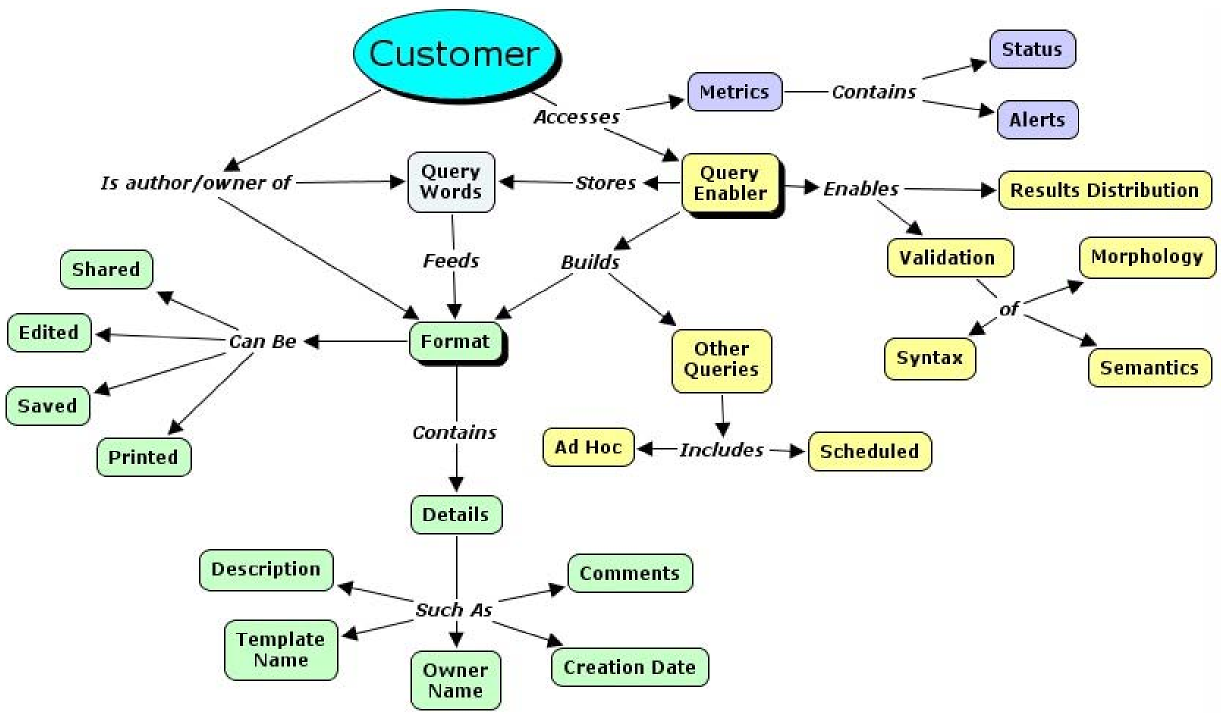

Figure 6: CMap of a Fabricated Vendor Deliverable of a Query System

John L. BeVier and Colleen A. Calimer 2004

On reason that Cmaps work so well in the Systems Engineering/Customer domain is inclusion of "cross-links". "These are relationships (propositions) between concepts in different domains of the concept maps. Cross-links help us to see how some domains of knowledge represented on the map are related to each other." (Novak). Cmaps allows several concepts to be displayed at the same time. Therefore you can depict the system functions AND the customer needs simultaneously.

Color-coding emphasizes the various components or subsystems and allows the reader's eyes to follow a natural flow for understanding. It also allows the reader to immediately disregard areas that are not of interest. Symbols, shapes, shadows and colors aid in clustering the concepts as the Cmap is being created.

Cmaps and Program Management: In the experiment, we captured the status of the TO BE system by color-coding the progress that the developer had made thus far, as well as the developmental priorities of the work yet to be done, as shown in **Figure 7.**

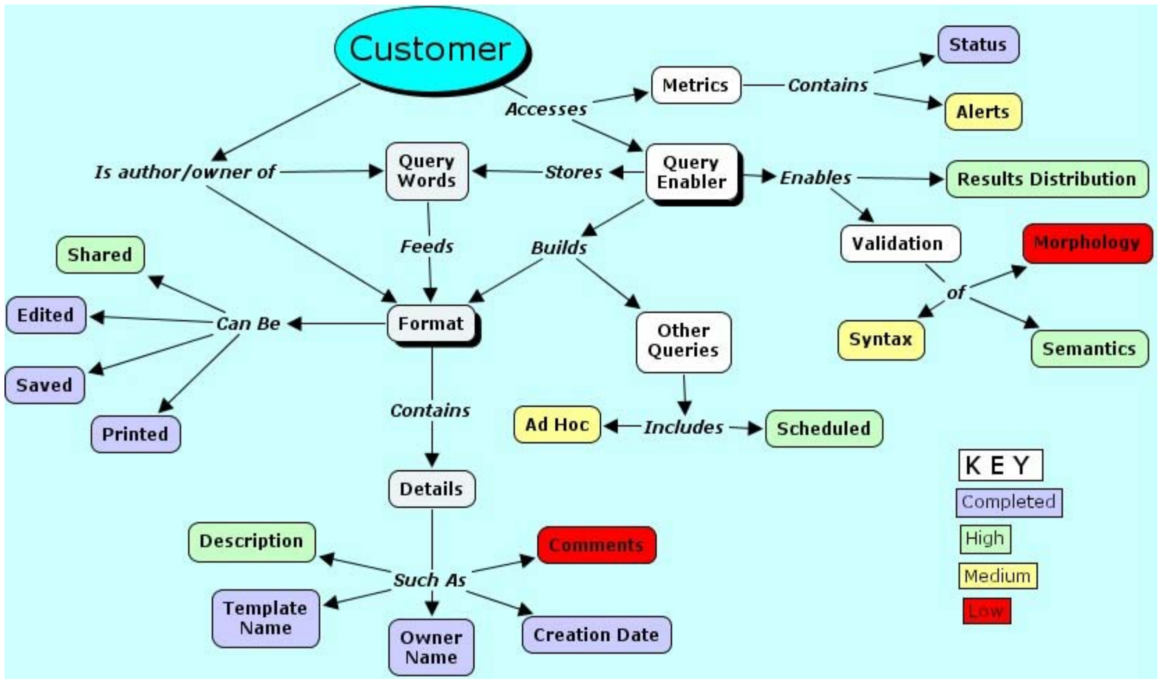

Figure 7: Snapshot: Vendor Priorities in Deliverables for Operational Assessment

John L. BeVier and Colleen A. Calimer 2004

The vendor provided the order in which they were going to develop the components of the system, as well as that which was already developed. The purpose was for understanding in each of the domains: the customer domain and the system domain. . The components yet-to-be developed were color coded into High (green), Medium (yellow) and Low (red) priorities. If there was to be any slippage, the customer was able to provide input so that the engineers could adjust their development efforts accordingly.

For instance, the developer thought that metric alerts were of little consequence to the customer and it would take away resources from other priorities. With Cmap-enabled dialogue, the customer readily saw the yellow/medium priority of the alert system application and could point out that alerts acted as "first responders" to their overloaded bin of query responses. Alerts focused the customer's attention to their highest priority mission requirements and were extremely important; more important than some of the other so-called high priorities. In this way, both the customer and the engineers were empowered to show the impact of these kinds of developmental decisions in each of their respective domains.

In the same vein, the systems engineer could explain the trade-offs of applying resources to the various system components and the impact it would have on other components. They were able to understand each other's priorities and constraints as the developmental effort revealed them, and as the Cmap dialogue reflected the state of the system accurately and with great clarity. This process was unique but created a valuable synergy between the builder and the buyer.

Cmaps and Operational Assessment:

The experiment included sharing the developer's intentions with an independent operational assessment (OA) organization. The OA organization was unfamiliar with the customer's domain as well as being unfamiliar with systems engineering developmental efforts. With the Cmap of the deliverables, the OA team was able to understand what the customer would be expecting (and what might not be delivered on time). The OA team was empowered with systems engineering and customer knowledge all in one diagram. They were able to create benchmarks against which to test. The OA team sent a letter of appreciation in reaction to the Cmaps and indicated that this would become the new way to build their OA tests in the future.

Cmaps and Training:

Another important communication value is the ability of the engineer to clearly articulate the steps and processes necessary for the customer to effectively use the TO BE system; i.e. training. Processes (what to do) and concepts (why to do it) can be expressed in a Cmap. Additional information, including help tabs, can be added to each object in the Cmap. Training can be accomplished with hands-on explorations of the actual system as well as with hands-on explorations of the virtual system in the form of the Cmap.

Cmaps Enhance Other Systems Engineering Processes:

By using the process of active communication enabled by concept maps, the entire requirements process can be updated so that more brainstorming by both engineer and customer can occur in the beginning of the project. As stated in many engineering books, the requirements gathering process has to be done right in order to avoid problems down-stream.

As the developer gains knowledge of the customer's domain and the “real” requirements, Cmaps can be used to explain the systems engineering constraints and risks to the customer. This can help with customer expectations against the backdrop of reality. And the customer can continue to drive the requirements process throughout the Systems Engineering process instead of letting go of the reigns once the requirements are captured and documented.

Cmaps can be used to jump-start the Systems Engineering processes. Cmaps, used in the requirements gathering process to capture and understand the domain of the customer, documents the AS IS system. These Cmaps should be used as a basis to inform the construction of Cmaps of the TO BE system. The TO BE system is defined as the processes that achieve the desired business objectives, giving rise to new business rules and data elements.

Depiction of the system as a set of interacting concepts at the Cmap level is normally not precise enough for the design and engineering processes, therefore Use Cases may be required from the Cmaps of the TO BE system, especially for new business rules and data elements.

Summary

The Cmap speaks for itself: See **Figure 8.**

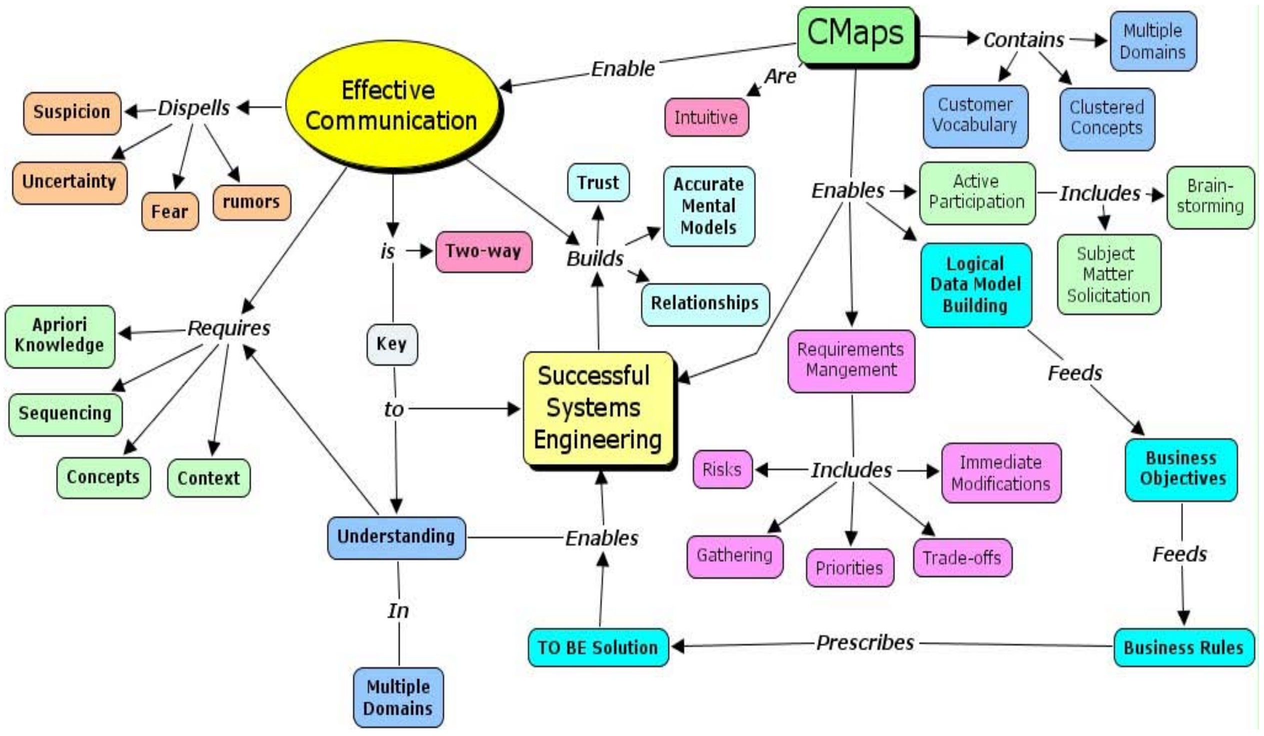

Figure 8: Summary John L. BeVier and Colleen A. Calimer 2004

REFERENCES

Ausubel, D. P. (1963). *The Psychology of Meaningful Verbal Learning*. New York: Grune and Stratton

Ausubel, D. P (1968). *Educational Psychology: A cognitive View*. New York: Holt, Rinehart and Winston.

Ausubel, D.P., J.D. Novak, and H. Hanesian. (1978). Educational *Psychology: A Cognitive View*, 2nd ed. New York: Holt, Rinehart and Winston. Reprinted, New York: Warbel & Peck, 1968.

Joseph D. Novack - *The Theory Underlying Concept Maps and How to Construct Them*.http://cmap.coginst.uwf.edu/info/printer.htlml

BIOGRAPHIES

Colleen A. Calimer, Boeing Inc., served in various positions and as a Senior Intelligence Analyst at the National Security Agency for 27 years until her retirement in December 2001. She then joined Conquest Inc. (later acquired by Boeing) as a Senior Intelligence Analyst in a SETA capacity responsible for intelligence analysis requirements designed to transform NSA operations into a more modern, efficient, and customer-centric agile enterprise. Ms. Calimer co-authored *World Class Intelligence Analysis*, presented in Washington DC, 2003 and *Embedded Systems Engineering* presented in Toulouse France, 2004. Ms. Calimer contributed to Vitech's construction of their 2002 INCOSE presentation, *Systems Analysis: A Tool to Understand and Predict Terrorist Activities*, 2002. Boeing Inc. encourages application of systems engineering principles to intelligence analysis. Boeing also encourages Colleen's training and exploration of the field of systems engineering for intelligence systems enhancements and transformation. Boeing has also charged Colleen with enhancement of their participation in INCOSE.

John L. BeVier, Principal of John L. BeVier & Associates, LLC, graduated Ohio State University, B.S. in Mathematics and minor in Physics. Served the National Security Agency for 40 years in computer programming, systems analysis, Branch and Division management, Program Management, and Senior Technical Staff Chief positions; Program Manager of various sizes from up to $125M per year and served as Requirements Chief of a Program with a greater than $1B per year budget. Became the first NSA point of contact for Military Support to Tactical Exploitation of National Capabilities, TENCAP. Retired Apr 2001 to operate his consulting business that applies the engineering and intelligence analysis techniques described in his earlier co-authored papers: *World Class Intelligence Analysis*, presented in Washington DC, 2003 and *Embedded Systems Engineering*, presented in Toulouse France, 2004.