Language Data – The 'Other Data' in the Six Sigma World

David Hallowell
Managing Partner
Six Sigma Advantage, Inc.

This two part series will focus on the kind of data that gets less press than the numbers so visible in our Six Sigma work – Language Data.

Part 1 will consider:

- The role of language data in our Six Sigma work
- The nature of language data – its types and refinement
- Tips for gathering better language data

Part 2 will cover some useful ways we can process and use language data, considering:

- Pre-processing raw language data for specific uses
- Focusing on an efficient data sample
- Distilling the data for particular purposes using tools like:
  - Net-Touch
  - KJ
  - Affinity Diagrams

Isn't Six Sigma About Numbers?

Everyone involved in Six Sigma, from the leaders and champions through the belts and team members, learns the importance of fact-based communication in both directions: (1) Input = listening, gathering facts, distilling, and (2) Output = reporting, convincing, and motivating. Six Sigma has done so much to improve the way that quantitative and graphical tools strengthen communication it would be easy to think that numbers must be the key to success. In practice, though, we know that the numbers without the 'story' won't drive all the necessary learning and reporting over the course of most DMAIC or DFSS projects. Especially as the number of people involved increases (Figure 1) we need to gather and broadcast better language data in order to succeed.

Another fundamental drive to better understand language data is built into the proactive and reactive problem-solving processes underlying DFSS and DMAIC. During project selection or the early stages of many projects, before we even get to the numbers we have to **formulate** the problem or opportunity using the information most available from customers, stakeholders, and the target environment. That information is largely in the form of language data (Figure 2). .

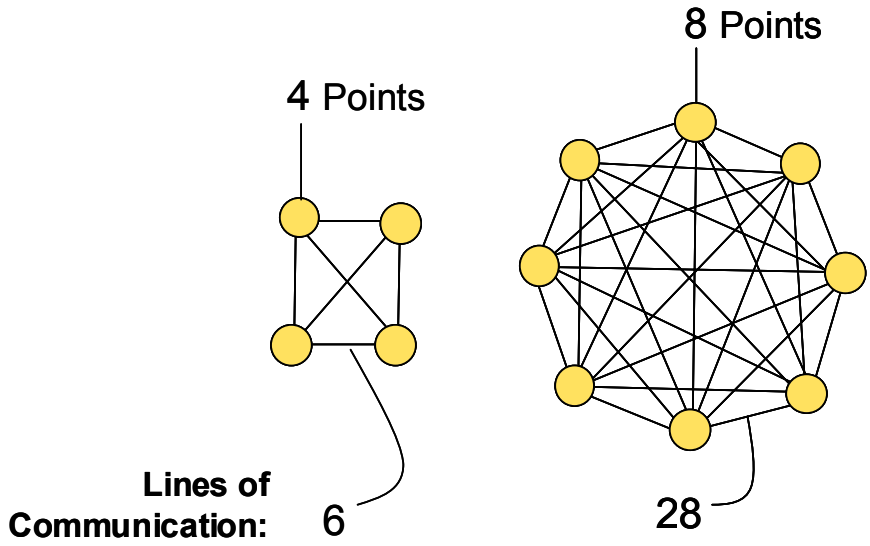


Figure 1
Points of Contact Increase Communication Risk

A few people, working closely together on the same project may be able to manage communication rather informally to keep in sync. As the number of points of contact (n) grows (through the size of the team, stakeholders, customers, suppliers, etc.) the number of lines of communication grows dramatically as the number of two-way combinations, or $\binom{n}{2}$ The varied perspectives, motivations, and ability to focus across the lines in any typical project suggests that communication cannot be left to informal means. ’

Figure 2 illustrates the use of data to drive progress in problem-solving. This builds on Kawakita's 'W' model[1] and Shiba's 'WV' model [2] which have unique insights into the role of language data. The vertical axis depicts an individual or team moving back and forth between 'Thinking' - where they reflect on data previously acquired, distilling it to gain understanding - and 'Experience' - being immersed in the data at its source. 'Thinking' involves planning for the gathering of any new data that would be useful, and 'Experience' implies making the best use of limited time to gather the most useful and accurate data available.

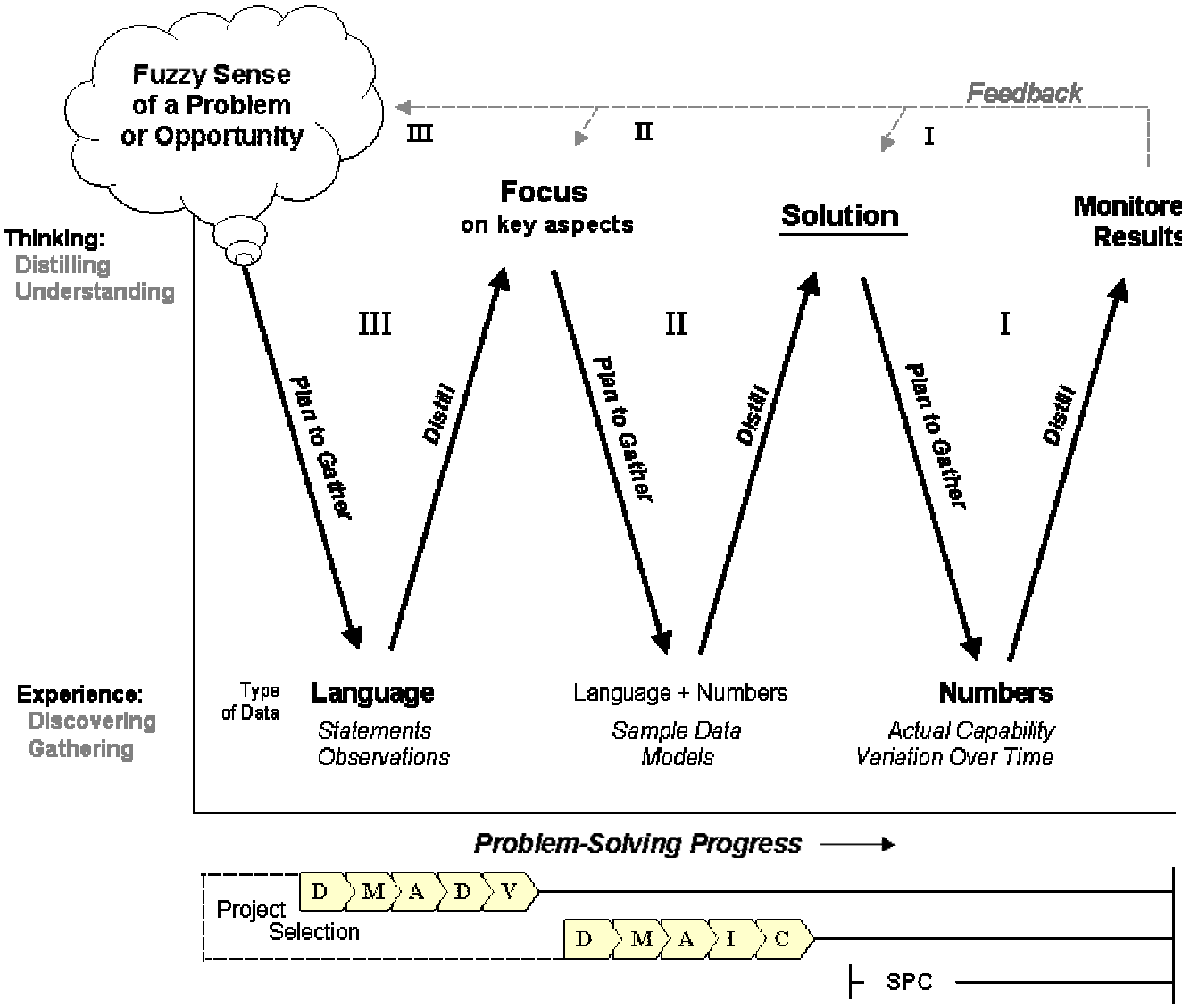


Figure 2
The Role of Language Data in Problem-Solving

A key point in Figure 2 is that, at the front end, language data is what we often have to work with. DMADV (DFSS) projects in the early stages, for example, deal with customer and business environment data that is predominantly statements about requirements and observations about a target environment ('needs and context',' as described below). Distilling that data helps a team to focus on the important aspects of the opportunity or problem in order to think about the next, more detailed set of data to collect.

The region labeled “II” is where a DMAIC project would typically start. DMAIC begins with a strong focus on some aspect of a potentially broader issue.

Indeed, if a DMAIC startup feels like there is only a ‘fuzzy sense of the problem’ the project is too broadly scoped and not ready to hand to a DMAIC team.

Statistical process control (SPC) begins with a 'Solution' and gathers data that is mostly numeric to provide monitoring and control feedback.

Dealing with Language Data 'Measurement' Variation

While language is different than numbers in many ways, there are some parallels in the sense that our data can be tainted with unwanted variation - sometimes imposed by our 'measurement system.' Language as we often find it, in its raw form like conversations and Email, is often colored with emotion, judgment, inference, and unclear measures. While these convey some meaning, they superimpose a lot of noise, making the real meaning ambiguous or unclear. To improve the accuracy and repeatability , Hayakawa[3] distinguishes the value of 'report language' - which focuses on the traceable facts (Figure 3).

As problem-solvers we can generally operate a lot better when our qualitative data is gathered and processed in the form of 'report language.' That doesn't always come easy, as people tend to generate the emotion, judgment, etc. at the top of Figure 3. It takes energy, usually effective probing with follow-up questions, to acquire the data we need.

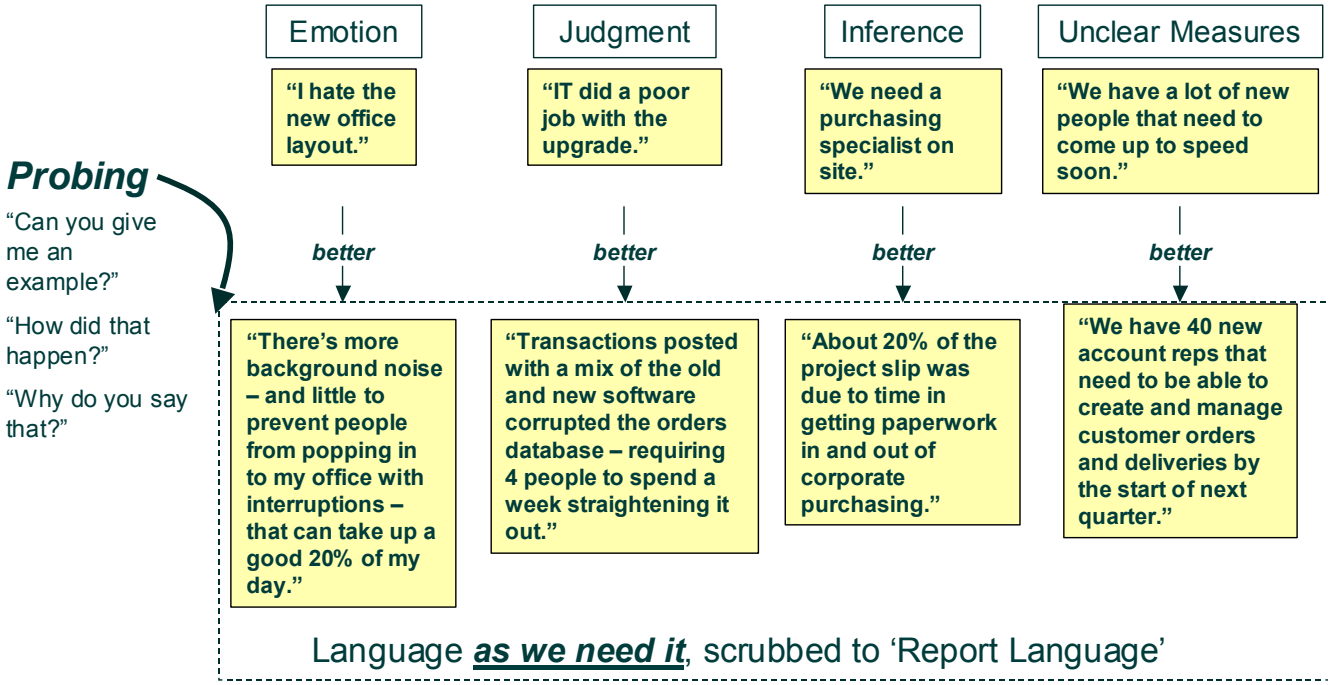


Figure 3
Removing 'Measurement Noise' During Language Data Acquisition

Abstraction

Without giving it a name, people routinely exercise the powerful skill of distilling the common aspects in a number of specific events or things, classifying them at the appropriate detail layer in their ever-growing mental database. This is, of course, abstraction, which is a core communication and thinking skill. Hayakawa [3] developed the useful notion of the 'ladder of abstraction' (Figure 4) which helps us to appreciate the abstraction isn't about being 'vague' but rather it involves 'precise generalization - finding the right 'rung' that has enough detail to be clear, without so much that it gets in the way. It would be impossible to converse or think without our seamless ability to move up and down that ladder . We don't say "I'm going to the apple, banana, grape, strawberry stand," when 'fruit stand' conveys the idea with appropriately less detail. On the other hand, we don't ask someone to bring us back 'some food' for lunch. While these examples seem trivial they show how automatic abstraction is for us.

Abstraction is easy to see in the classification of simple objects -- and a little more subtle when it comes to organizing phrases or statements. Figure 3 shows the same general idea at three levels of abstraction. Part of our challenge is

knowing which level is right for a particular use. We can see that higher levels express themes and concepts while lower levels show examples and illustrations.

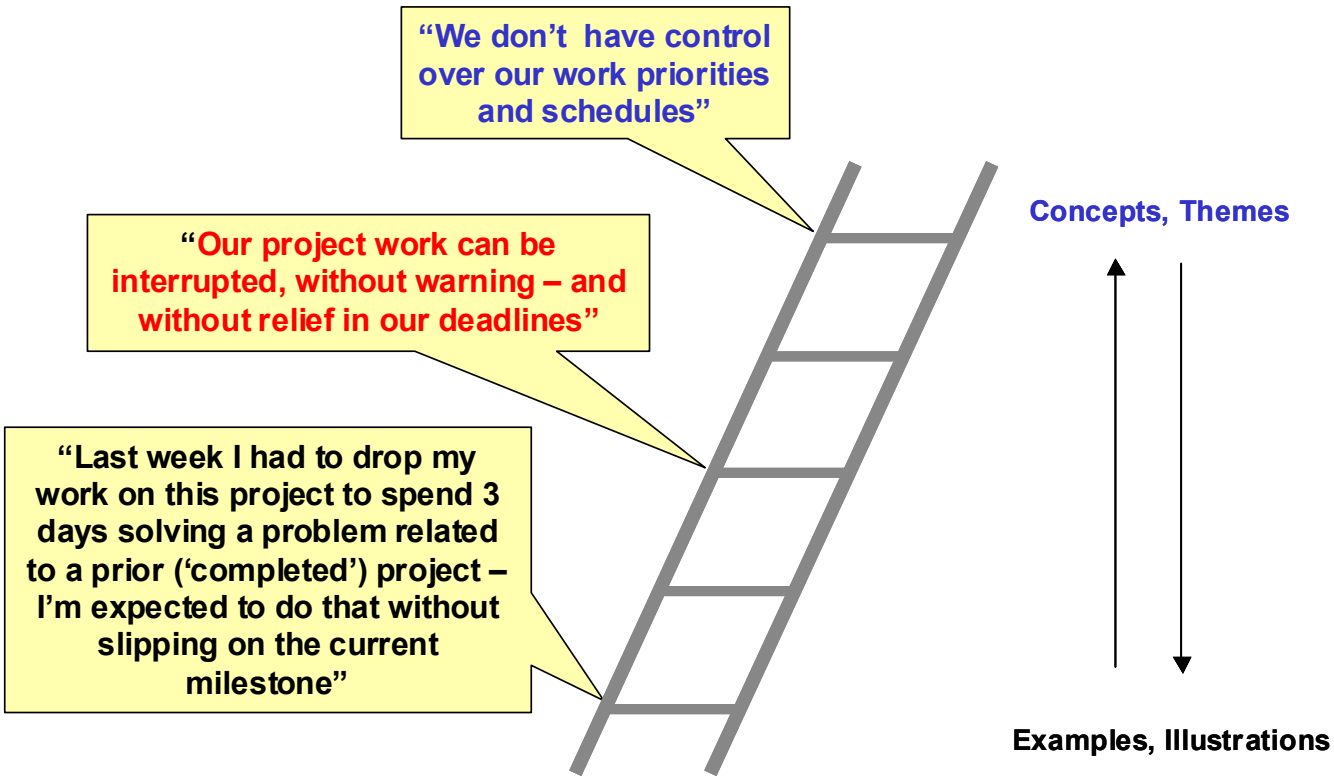


Figure 4
The Ladder of Abstraction

Context and Needs Data

Noting the distinction between what we'll call 'context data' and 'needs data' can help us gather better data and it can change the way we process that data. We We're already familiar with 'Needs' - as language that conveys something that a would be valuable to have or be able to do. We gather that data by asking or watching things like 'What would you like to be able to do?" 'Context,' on the other hand, will refer to observations or statements about an environment and the things that go on there (or might in the future). We gather context data by

asking or observing things like “How do you do that?” or “Describe that situation.” In Part 2, when we look at tools like the KJ and use cases we will see how each of these kinds of data carry important information that can help us formulate a complex problem, develop stated and latent requirements, or to discover factors important to robust design. Companies that understand the value that is hidden in context data pay a lot of attention to gathering and processing it well.

Focused Discussions for Better Language Data

When we set out to gather language data that’s as good as it can be by all the above criteria: right level of abstraction, report language if appropriate, the right mix of useful context and needs data – one tried and true approach is the focused, yet open-ended interview. The trick is to steer the interviewee into discussion areas that provide data that supports the learning most useful to the interviewer – without making them feel they are being ‘grilled.’ As illustrated in Figure 3 moving from data ‘as we find it’ (often an interviewee’s first responses) to the level of detail and clarity that we’d like, usually involves probing. Mastery is reached when probing comes across to an interviewee as a natural expression of an interviewer’s genuine interest and empathy. Since much of the richest data tp be harvested from an interview will be found in the answers to follow-up (probing) questions and remarks – fledgling interviewers would be wise to focus on learning and practicing that skill. More detail and many useful tips in this area can be found in McQuarrie's classic text 'Customer Visits [4].

Capturing good notes and/or recorded transcripts is also important. Small minidisk recorders and unobtrusive microphones are recommended for recording. Video and ‘typing notes into a laptop during an interview are not recommended and are sometimes prohibited by security or other considerations.

Looking ahead to Part 2

Having looked at our motivation gathering better language data and some aspects of its ‘nature’ we will focus in Part 2 on some useful tools we can employ for language data processing.

References

[1] Jiro Kawakita. ***The Original KJ Method***. Kawakita Research Institute. 1991

[2] Shoji Shiba et al. ***The Language Processing Method***. CQM. Boston. 1995.

[3] S.I. and Alan R. Hayakawa***. Language in Thought and Action***. Fifth Ed. Harcourt Brace Jovanovich, 1990

[4] Edward F. McQuarrie. ***Customer Visits***. Sage Press. Second Ed. 1995.

Language Data – The 'Other Data' in the Six Sigma World
Part 2

David Hallowell
Managing Partner
Six Sigma Advantage, Inc

Part 1 focused on the nature of language data and its place in the broader Six Sigma toolkit. Part 2 will cover some useful ways we can process and use language data, considering:

- Pre-processing raw language data for specific uses
- Focusing on an efficient data sample
- Distilling the data for particular purposes using tools like:
  - Net-Touch
  - KJ
  - Affinity Diagrams

Pre-processing Raw Language Data

One problem with language data is the volume of raw material we have to work through in order to mine the most useful material. For voice of the customer data it can be helpful to first pre-process transcripts to highlight context and needs data, as discussed in Part 1. Figure 1 illustrates this for a simple case. Extracting key phrases, while maintaining a clear, traceable, link to their sources, is the first step in preparing the data for further distillation.

Customer – a warehouse manager

I: What kinds of data does the system need to interact with?

C: We need to be in constant touch with out MRP system and the related inventory files. Orders incoming from sales are read from systems in either of three servers around the world. (see the attached interface sketch)

I: Are orders coming in around the clock?

C: We hope. Keeping up with the follow-the-sun timing is challenging. Delays in our ability to confirm availability and delivery create problems for sales … and of course we hear about that right away.

I: So your inventory, status, and planning information needs to keep pace through all three shifts.

C: That's right – and if as if that weren't enough – the systems at our different sales sites and design centers are at various levels of capability and standardization. We are into the third month of a global update – but for another three months we will have to talk to a mix of the old and new systems and translating back and forth between them.

I: When that stabilizes will things get easier?

C: That would be nice. But then it will be a new supplier system or some other database we need to write a new interface for.

I: How do you plan the AGV routes and schedules?

C: First the system has to gather up all the data about the production plans, the availability and locations of parts, and the location, capacity, and availability of each of the workcells. With that the system can connect the dots to be sure all the needs are covered.

I: That has to be a pretty complex problem – figuring the best route.

C: It is – but our current system doesn't necessarily compute the best route – if you can come up with a better way, we're interested. We can't quantify it – but our vehicles must be wasting some meaningful time and energy waiting to pick things up or looping back to pick or drop things.

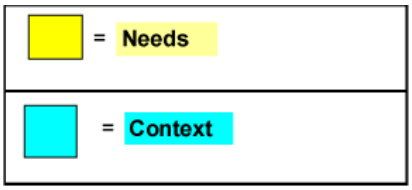


Figure 1
Pre-processing transcripts to highlight context and needs data

Focusing on an Efficient Data Sample

Even after some pre-processing we may simply have too much volume of data to readily distill. Some further pruning to a representative data sample can save a lot of time and energy. When we discuss the KJ method, for example, we'll see that its high level of discipline calls for considerable time spent on each language data element introduced. Experience shows that about 20 to 25 such data elements, with the right detail level and scope, is just about the right count for the inputs.

A Multi-Pick Method

What if our pre-processing has yielded 100 or more language nuggets? A simple approach that works very well in practice works as follows (Figure 2).

1. On an empty wall chart, post the 'theme' question that the team is trying to answer with language data. This provides an important focal point for deciding what's most useful and what could be trimmed.
2. Place each element of the data set on a separate self-stick note, arranging the notes on the empty chart.
3. Each team member will read each note, considering whether or not that element should be considered further as part of the critical sample.
    - 'Survival' of notes toward the critical sample is designated by their accumulating red dots in their lower right corner - but note this is not a voting process. During any round, the most dots a note can acquire is the same as that round number (for example, in round one, notes start with no dots and each may or may not acquire one dot).
    - As the reading and marking go on in parallel, some readers will come to a note that already has been marked to survive the current round. In that case, the instruction is to read the note, recognizing it as a survivor. This creates a kind of progressive filter in the reader, who may be more discriminating about applying dots to other notes that are similar.
    - There comes a point of diminishing returns, where the arrival of new dots slows way down. At that point the facilitator will separate the surviving notes and stack the others as 'takeaways' for that round. These will be kept as part of the documentation - sometimes yielding one or more useful notes after some reconsideration downstream.

Distilling The Data

Net-Touch

This is a simple affinity process, with a useful twist In a routine affinity, everyone's self-stick notes are posted on a wall and each has to be read by enough of the group to get the grouping started In Net-Touch, everyone holds

onto their own notes and watches the facilitator for the cues to offer a note for grouping. A good practical use for this process is during the building of an interview discussion guide. These steps should illustrate the way it works:

1. Everyone in the group writes open-ended questions that address the VOC learning objectives
2. People hold onto their own notes.
3. A facilitator takes one note from the group, at random.
4. Reading that note, they ask - "Does anyone have a question that belongs with this one?"
5. Because everyone is familiar with their own notes (and because they haven't had to take the time read anyone else's), this is efficient.
6. The facilitator collects all the notes that are offered, forming a cluster with the original, seed note.
7. Repeat steps 3-6 until all the notes have found their way into a cluster.
8. Now the group can work together to 'clean up' (subdividing large clusters to smaller groups, for example) and 'title' the groups.

KJ Analysis

Because the KJ is a powerful tool that is not widely used, we'll give it some special attention here. Its founder, Jiro Kawakita, realized the simple yet profound value in the way that abstraction distills meaning, even in language and observational data that is incomplete.

There are many kinds of KJs - each distinguished by the theme question posed in its upper left corner (see Figure 3 and the example KJ in Figure 4). Important to note about all KJs is that they seek facts that answer the theme question. For that reason, you won't typically see a KJ used for 'brainstorming' - at least in the sense of idea generation. Part of the discipline in KJ is the use of 'report language' and the verification its source data.

KJ Type	Theme	Data	Uses
Context / Image	**What scenes and images describe …?**	**Word-pictures** "The (secure) network password is plainly taped on some monitors"	**Understanding an "environment"**
Requirements	**What are the key requirements for ..?**	**Needs** **(solution-free)** "System verifies item availability"	**Finding themes and underlying messages in a complex set of needs**
Weakness / Problem-Formulation	**What are the key problems ….**	**Facts** "Our developers are interrupted – without warning – to work on support issues about 20% of their time."	**Formulating a problem. Focusing on where to do more detailed problem-solving work**

Figure 3
Types of KJs, Their Data, and Their Uses

KJ Steps
The only way to really learn to do a KJ is to do one or a few under the guidance of someone who knows the 'rules' and has had some practice. The steps are outlined here as a guide to what is involved.

1. **State the THEME (a question - per FIgure 4)**
The theme carefully states what the team seeks to learn with the KJ. As such, it is the 'hook' that invites all the data elements to be brought as factual answers.

2. **Gather facts (that answer the THEME question):**
A color-code convention suggests that these facts be printed in black, usually on self-stick notes.

3. **Assemble the facts, reduce to the key 20-30 if necessary**
See the Multi-Pick method above and in Figure 2.

4. **(Team) Understand each fact, scrubbing its language for clarity**
A powerful step in the process - has each fact being explained by the person who brought it. The team comes to understand the story behind and around that fact and they work together to be sure it is stated clearly. Even if they stopped here, a team would probably have derived significant benefit by

having come to understand one another's facts and perspective on a complex issue.

5. **Group the facts that really belong together (3 per group, maximum): Lone wolves very OK**

This is an important part of the discipline - notes are not grouped by 'keyword' or 'logical' grouping (the way an engineer might) but by the "story they tell." When in doubt, it is often better to leave a note out of the grouping (as a lone wolf) because a foreign element dilutes the abstraction that is so important in creating the next level up in the KJ structure.

6. **Title the groups: RED. Complete sentences, follow abstraction 'is-a' rules**

Every piece of data on a KJ is a complete-sentence answer to the theme question. This is true of the 'titles' that the team creates to describe each group at each level. Note the red and blue titles in Figure 4. The trick, at this step, is to write a title that is as specific as possible, while generalizing the stories being told by each of the notes in the group (that 's the 'precise generalization' sense of abstraction followed here). In object-oriented design a common test of an abstraction hierarchy Is that any item below is a reasonable substitute for an item above it - in shorthand referred to as an 'is-a' relationship (a collie 'is-a' dog which 'is-a' pet which 'is-a' animal).

7. **Encapsulate lower level facts under the RED titles**

Simply hide the black notes under the reds - to simplify next-level grouping.

8. **Group (3 max) the RED titles and any Lone Wolves that really belong together**

Repeat the logic and operations of step 5.

9. **Title the groups: BLUE**

Repeat the logic and operations of step 6 -at this higher level of abstraction.

10. **Encapsulate the RED groups under BLUE titles**

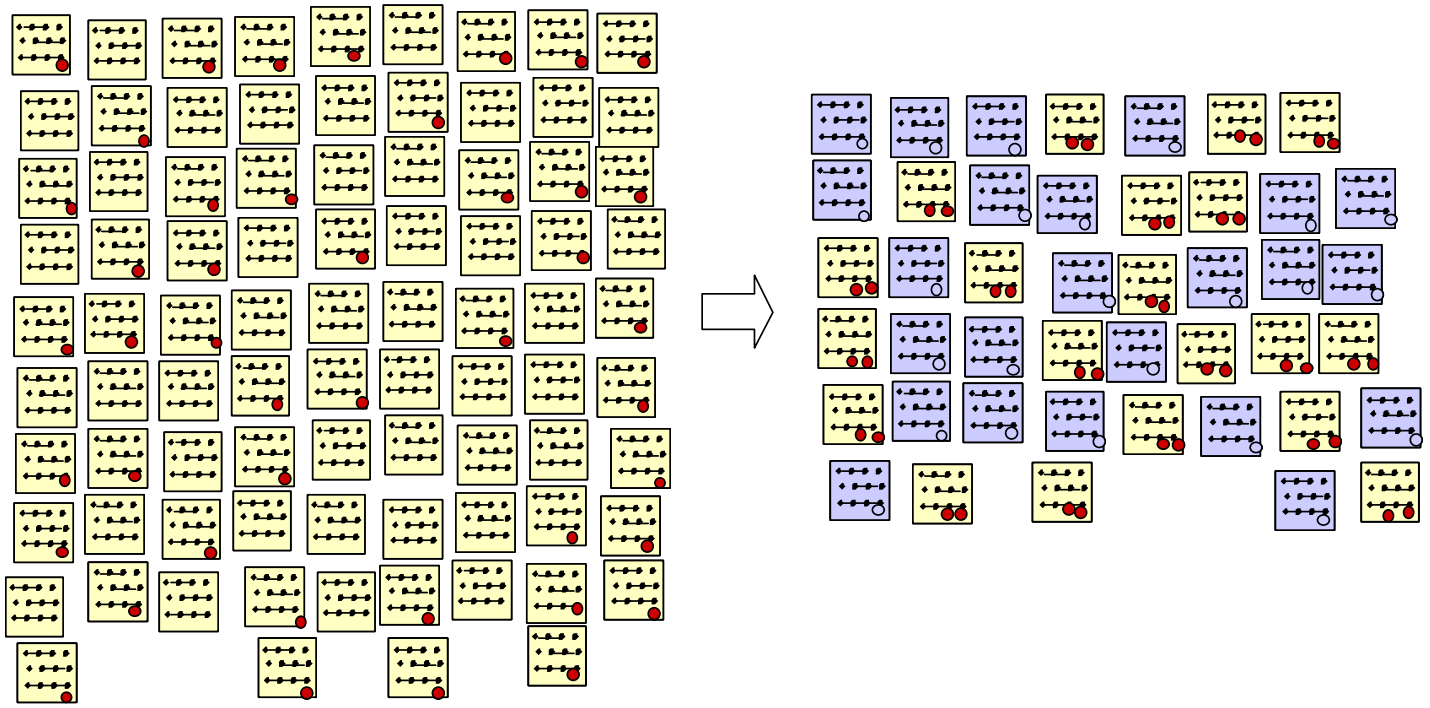
Like step 7.

11. **Arrange the BLUE Titles and any remaining RED or Lone Wolf items to describe cause and effect relationships.**

    Use arrow and 'opposition' symbols (not always done on Context KJ)

12. **Vote on the most important RED level groups or lone wolves**
13. **Write a CONCLUSION statement. Upper right, in RED**
14. **Sign and date the KJ**

WHAT SCENES AND IMAGES
DESCRIBE THE ____TARGET ENVIRONMENT ?



Round 1

- Starts with Original, Unmarked Notes (85)
- Ends with Single Dot Selections (44)

Round 2

- Starts with 1-Dot Survivors
- Ends with 2-Dot Selections (20)

Figure 2: Focusing on an Efficient Data Sample

Figure 4 A 'Problem Formulation (Weakness-based) KJ