Applying the Fishbone diagram and Pareto principle to Domino

Part 1

Dhanasekar Dhandapani

June 21, 2004

The Fishbone diagram is a simple problem-analysis tool that you can use to analyze Lotus software-related issues. This article, part one of two, introduces you to the Fishbone diagram and as an example, applies it to an actual Notes/Domino issue.

In any organization problem analysis and management tools are crucial to success. In software quality management, there are two tools that you may want to make use of: the Fishbone diagram and the Pareto principle. In this two-part series, we introduce you to the problem analysis tool known as the Fishbone diagram and to the management principle known as the Pareto principle. We discuss how these techniques are relevant to Notes/Domino and how to use them through examples. The purpose of the Fishbone diagram is not to find solutions to Notes/Domino-related problems, but to determine the root cause--module, design element, or application--of a problem. The article is intended for experienced Notes application developers and Domino administrators with little or no knowledge of the Fishbone diagram.

About the Fishbone diagram

The Fishbone diagram is also known as the cause and effect diagram, the root cause analysis, and the Ishikawa diagram, named after its originator Kaoru Ishikawa, the Japanese quality pioneer. It is generally called the Fishbone diagram because the diagram resembles that of a fishbone. In simple terms, Fishbone is brainstorming in a structured format. The technique uses graphical means to relate the causes of a problem to the problem itself, in other words, to determine cause and effect. The diagram focuses on the causes rather than the effect. Because there may be a number of causes for a particular problem, this technique helps us to identify the root cause of the problem in a structured and uncomplicated manner. It also helps us to work on each cause prior to finding the root cause.

This technique is very much applicable to the software industry and to Notes and Domino. There are problems in Notes-based applications and in Domino administration in which root cause analysis is important. For example, replication problems can occur for a number of reasons,
including replication settings, database access levels, document security, or other factors. The Fishbone diagram helps us to arrive at the root cause of a problem through brainstorming.

**When to use it**
You may find it helpful to use the Fishbone diagram in the following cases:

- To analyze and find the root cause of a complicated problem
- When there are many possible causes for a problem
- If the traditional way of approaching the problem (trial and error, trying all possible causes, and so on) is very time consuming
- The problem is very complicated and the project team cannot identify the root cause

**When not to use it**
Of course, the Fishbone diagram isn't applicable to every situation. Here are a just a few cases in which you should not use the Fishbone diagram because the diagrams either are not relevant or do not produce the expected results:

- The problem is simple or is already known.
- The team size is too small for brainstorming.
- There is a communication problem among the team members.
- There is a time constraint; all or sufficient headcount is not available for brainstorming.
- The team has experts who can fix any problem without much difficulty.

**Relevance of the Fishbone diagram to Notes application support**
Most of you have experience in supporting and administering Domino. You have probably solved problems ranging from the simple to the complex that take anywhere from a few minutes to hours or even days to resolve. For the problems that stumped you most, you may have approached your colleagues, friends, technical architects, or others for help. You might even have uncovered a lot of potential causes for a problem without knowing their actual relevance to the problem context, and you might have analyzed each and every one of them. This can lead to increased time taken to find the root cause of the problem.

Using the Fishbone diagram, you can approach the same problem in a more systematic and uncomplicated manner. After listing the possible causes for a problem, you and your team analyze each one carefully, giving due importance to statements made by each team member during the brainstorming session, accepting or ruling out certain causes, and eventually arriving at the root cause of the problem. In general, Fishbone diagrams give you increased understanding of complex problems by visual means of analyses.

**How to construct a Fishbone diagram**
Here are the various tasks involved in constructing a Fishbone diagram:

1. Define the problem
2. Brainstorm
3. Identify causes

**Define the problem**
The first step is fairly simple and straightforward. You have to define the problem for which the root cause has to be identified. Usually the project manager or technical architect--we will refer to this role as the leader throughout the rest of the article--decides which problem to brainstorm. He has to choose the problems that are critical, that need a permanent fix, and that are worth brainstorming with the team. The leader can moderate the whole process.

After the problem is identified, the leader can start constructing the Fishbone diagram. Using a sheet of paper, she defines the problem in a square box to the right side of page. She draws a straight line from the left to the problem box with an arrow pointing towards the box. The problem box now becomes the fish head and its bones are to be laid in further steps. At the end of the first step, the Fishbone diagram looks like:

**Figure 1. Fishbone diagram, Step one**

![Fishbone Diagram](image)

**Brainstorm**

People have difficulty understanding how to structure the thought process around a large problem domain. Sometimes it is useful to focus on logically related items of the problem domain and to represent them in the Fishbone diagram, which will convey the problem solving methodology. There are quite a few tools available that can help us in this regard, including:

- **Affinity Chart**
  Organizes facts, opinions, ideas, and issues into a natural grouping. This grouping is in turn used as an aid in diagnosing complex problems.

- **Brainstorming**
  Gathers ideas from people who are potential contributors. This process is discussed further in the following sections.

- **Check sheet**
  Acts as a simple data recording device that helps to delineate important items and characteristics to direct attention to them and verify that they are evaluated.

- **Flow charts**
  Organizes information about a process in a graphical manner and makes it clear who is impacted at every stage.

No single methodology is applicable to all problem domains. Based on experience and study, you can identify, thoroughly analyze, and maintain the methodology and the related problem domains. In the example given later in this article, we use brainstorming as the problem solving methodology.

**Categorize**

When you apply the Fishbone technique to business problems, the possible causes are usually classified into six categories:
• Method
• Man
• Management
• Measurement
• Material
• Machine

Though the above are a few important problem categories, during the brainstorming session, the team is encouraged to come up with all possible categories. The above categories give the team direction to help find the possible causes. Some of the categories listed above may or may not be applicable to software or to Domino in particular. Let's look briefly at each category.

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method</td>
<td>Methods are ways of doing things or the procedures followed to accomplish a task. A typical cause under the Method category is not following instructions or the instructions are wrong.</td>
</tr>
<tr>
<td>Man</td>
<td>People are responsible for the problem. The problem may have been caused by people who are inexperienced, who cannot answer prompted questions, and so on.</td>
</tr>
<tr>
<td>Management</td>
<td>Management refers to project management; poor management decisions, such as upgrading two components simultaneously rather than deploying changes serially may cause technical problems.</td>
</tr>
<tr>
<td>Measurement</td>
<td>Measurement refers to metrics that are derived from a project. Problems may occur if measurements are wrong or the measurement technique used is not relevant.</td>
</tr>
<tr>
<td>Material</td>
<td>Material basically refers to a physical thing. A bad diskette is one typical example. Software can't always handle errors caused by bad material, for instance a bad backup tape, so while material may be the least likely cause, it is a possible cause.</td>
</tr>
<tr>
<td>Machine</td>
<td>A machine in software usually refers to the hardware, and there are a lot of possibilities that a problem can be due to the machine, such as performance issues.</td>
</tr>
</tbody>
</table>

Other possible categories include policies, procedure, technology, and so on.

After identifying a problem, the leader initiates a discussion with the project team to gather information about the possible causes, finally arriving at the root cause. The team can either analyze each of the above categories for possible causes or look into other categories (not listed above).

**Identify causes**

While brainstorming, the team should strive toward identifying major causes (categories) first, which can be further discussed, and then secondary causes for each major cause can be identified and discussed. This helps the team to concentrate on one major cause at a time and to refine further for possible secondary causes.
After the major causes (usually four to six) are identified, you can connect them as fishbones in the Fishbone diagram. They are represented as slanted lines with the arrow pointing towards the backbone of the fish. See Figure 2 later in this article.

Sometimes it is difficult to arrive at a few major causes. The team may come up with a lot of causes, which makes brainstorming more difficult. In this case, the leader can assign 10 points to each team member for each possible cause, and let them assign the rating (from 1 to 10, 10 being most likely cause) to each cause. After everyone on the team has rated the causes, the project manager totals each of the causes and ranks them based on their ratings. From the list, the top four to six causes are identified as major causes and connected as bones in the diagram.

The diagram looks a little like the skeleton of a fish, hence the name Fishbone. After the major causes of the problem are identified, each one of them is discussed in further detail with the team to find out the secondary causes. If needed, the secondary causes are further discussed to obtain the next level of possible causes. Each of the major causes is laid as a fishbone in the diagram and the secondary causes as "bonelets."

The diagram now has a comprehensive list of possible causes for the problem, though the list may not be exhaustive or complete. However, the team has enough information to begin discussing the individual causes and to analyze their relevance to the problem. The team can use analytical, statistical, and graphical tools to assist in evaluating each of the causes. The Pareto principle (explained in part two of this article series) is also used to find the elements that cause major problems and to list them as major causes in the Fishbone diagram. Software metrics that are obtained during application support can also be used here for further assistance.

Evaluate, decide, and take action

It may be very difficult to come up with consensus on a large team for one possible root cause, but the majority is taken into consideration. Also, the major causes can be ranked in order with the most likely cause at the top of the list.

After the evaluation process is complete, the action plan has to be decided. If one possible root cause is identified, then the action plan has to be derived to rectify it. Sometimes, it may be difficult to arrive at the root cause; there may be a few possible root causes. In this case, the action plan has to be drawn for each of the possible root cause.

After the action plan is ready, the leader can designate an individual or team to work on the plan and to rectify the problem permanently. If there are a few possible root causes, all the action plans are to be executed, and the most likely root cause is identified and fixed.

Example

The Fishbone diagram can be used to troubleshoot Domino administration and Notes application-related problems. Some complicated administration issues, such as SMTP mail routing, replication, server crashes, and so on, and application issues, such as database replication, can be better studied and analyzed using Fishbone diagrams.
Let's look at how to apply the Fishbone technique to find the root cause of a Domino server crash. The idea in this example is to explain how to apply the Fishbone technique rather than how to identify the cause of the crash because a crash may happen for a number of reasons.

**Define the problem**

In this example, the problem is already defined--Domino server crash. The team knows that the crash occurred because the server ran out of resources, but an analysis is still needed to determine why the resources are running low. The leader starts drawing the Fishbone diagram by mentioning the problem in the fish head as shown below. The time of crash, frequency of crash, and crash details are all gathered prior to the brainstorming session.

**Brainstorm**

The team now is involved in the brainstorming session to identify the root cause of the problem. We used the categories listed above as our starting point of discussion to identify the major causes first. We analyzed each category and their relevance to the problem. The following lists each of categories that figured in our discussion and whether it was accepted as a major cause.

- **Method**
  The method or way of doing things was considered by the team as a possible cause. Because programs written for the Domino environment using @formulas, LotusScript, Java, and so on may cause server overload (and therefore a crash), it was considered a major possible cause.

- **Men**
  The team discussed the possibility of people being the cause of the problem. It is accepted as a major cause because a few less experienced administrators handled the mail and application servers. Inexperience, negligence, and complacency are some reasons for mistakes that can eventually lead to a server crash.

- **Management**
  The project management team was not a possible cause because the issue here is more technical rather than managerial. The team unanimously ignored this category.

- **Material**
  The team also ignored this category because material is not relevant to the problem.

- **Machine**
  The team discussed the machine being a possible cause. Insufficient hardware configuration may be a problem leading to overloading and finally to server breakdown. The team accepted it as a major possible cause.

- **Technology**
  The team discussed the impact of the technology that has been used. Issues like anti-virus software, third-party tools, and software problem reports on the current Domino releases were pointed out, so technology was also identified as a possible cause.

- **Procedure**
  The team discussed the various procedures used within the Domino environment. Procedures used in user registration, application roll-out, and migration were analyzed. The team decided that procedure could not be a possible cause. The team thought that if procedure was the
reason, then the server might have crashed while executing the procedures. That didn’t happen, so this category was ruled out.

- **Policy**
The team discussed various policies used in the organization for the Domino environment. Policies are quite specific to every organization, so a problem in an organization due to policy is not applicable to other organizations. Policies like scheduled agents, their schedules, servers, and APIs are studied as potential causes. We considered policy as a major cause.

At the end of the preliminary brainstorming session, we constructed the following Fishbone diagram.

**Figure 2. Fishbone diagram, Step two**

All major causes identified above are connected as the bones in the Fishbone diagram.

**Identifying the causes**

The next step involved continuing to refine the major causes to find the secondary causes or the various causes occurring under each of the major categories listed above.

- **Method**
The team looked into the tasks (agents) running at the time of the crash. They analyzed the code of each of these tasks (agents) and studied their impact on server performance.
- **Men**
The team felt that the less experienced administrators may have overloaded the server by forcing replication or mail routing, which leads to higher workload on the server and eventually brings the server down.
- **Machine**
The machine was Windows-based and was sufficiently powerful to handle the workload. To date, no issue was reported of server inefficiency, and the administrators were confident that it was not the cause. The team decided to ignore this category, but kept it as the last option to revisit at.
- **Technology**
The administrators analyzed bugs/issues with the current release (in our case, it was Domino 6.5). They also considered other tools like Norton AntiVirus software and its release installed on the server machine and a third-party tool that makes a copy of all outgoing mail messages.

• **Policy**
  The team discussed agent scheduling, tasks that ran on the server, and C/C++/Java API codes used for specialized operations.

At the end of this session, all possible causes under each category were identified and connected as bonelets in the Fishbone diagram.

Rather than analyze individual secondary causes, the team stopped the brainstorming session at this stage. The team felt that sufficient information was available to identify the root cause of the problem. The team studied the time of crash, frequency of the crash, and the crash information stored in a file. At the end of the session, the Fishbone diagram looked like the following. Causes from each category were constructed as bonelets in the diagram. The team went about analyzing individual causes.

**Figure 3. Fishbone diagram, Step three**

Sometimes the Fishbone diagram can become very large because the team may identify many possible causes. This make the diagram complex-looking. A simple and neat-looking Fishbone diagram may indicate that a thorough brainstorming is not done or that the team lacks sufficient knowledge about the problem domain. A good Fishbone diagram is one which is complete and has explored all the possibilities for a problem. The team should identify all possibilities to arrive at a good Fishbone diagram.

**Method**
The team discussed the two points listed above. Agent coding was done in LotusScript and Java. The team agreed that Java coding was very minimal and done at a basic level, so it could not be
a problem. But they were suspicious about LotusScript code in the agents. It was identified as a potential cause.

Men
The team discusses with the less experienced administrators issues that occurred when the senior administrator was away, but none had occurred. The crash occurred while the senior administrator was present. Men were not the likely cause.

Policy
The schedule of the various agents was checked. There was no overscheduling of agents, so the server was not overloaded. A couple agents employed API code, but minimally. The agents ran without problem for the past year. The team assumed that these issues could not be potential cause.

Technology
The team analyzed the log of the anti-virus software and the third-party tool used, but nothing specific was reported. Also, the tools had been running without a problem for the past two years. The team assumed that technology could not be the potential cause. The Domino version was upgraded recently, and the team was suspicious about it. The team analyzed the SPRs/QMRs/QMUs for any reported bugs, but none were found.

Now the team was left with one important potential cause: the LotusScript code of the scheduled agent. The agent had approximately 5,000 lines of code involving lots of loops and checks. A FOR loop ran almost 5,000 times, and each time it ran, 100 IF statements were evaluated. This leads to a performance impact on the server machine and eventually caused it to crash. The team had identified the root cause.

Decide and take action
The team decided to change the IF statements to SELECT CASE. The code was modified and scheduled to run at the same time as before. This time, however, there were no server crashes. The agent completed successfully.

Conclusion
The above example details the application of the Fishbone diagram in identifying the root cause of a software problem. We hope that you find this method useful in diagnosing your own software problems. You can apply the Fishbone diagram to any number of issues that your IT organization encounters. In part two of our series, we cover the Pareto principle and how it can help you to manage problems that you determined using this analysis tool.
Related topics

- Read part two of article series, "Applying the Fishbone diagram and Pareto principle to Domino, Part 2."
- For more information about this topic, see the web page, "Fishbone diagram: A problem-analysis tool."

© Copyright IBM Corporation 2004
Trademarks
(www.ibm.com/developerworks/ibm/trademarks/)