Common IT Problems
Some Causes and Solutions

Linker Systems, Inc.
800-315-1175 — www.linkersystems.com — linker@linker.com

There are a number of problems companies run into in their Information Technology departments. We at Linker Systems have seen a lot of them. Below, we list the most common symptoms, causes we've seen, along with the typical solution to each problem.

The categories of the problems shown below are:

Speed:	All or part of the system is running too slowly.
Integrity:	The data in or coming from the system is unreliable.
History:	The system doesn't know what happened yesterday.
Sizing:	Data or programs are too large.
Networking:	The systems won't "talk".
Security:	Matters of data access, physical access, and control.
Work-flow:	Improving what you got the computer for in the first place.
Reality:	Time for a reality check.
Price:	Reducing your costs.

Speed

These problems all deal with system speed, or the speed of various components of the system.

Symptom: Keying data in or tabbing from field to field takes too long.

Possible causes	Solutions
The program may be performing queries against the data base at every step.	It should do queries only when absolutely needed.
The program may be validating data too early.	Any time-consuming validation should occur only at the last moment.

Possible causes	Solutions
The program may have been written in an interpretive tool, rather than a compiled one.	Try to speed it up as shown above and below. If that doesn't work, the program may have to be rewritten from scratch in something faster. Convert only the screens or forms needed, as this is an expensive fix.
The program may be written in Java, or some other "self-cleaning" language in such a way that "garbage collection" is happening too often.	Investigate the way "objects" or other memory structures are being allocated and released or dereferenced. Try to simplify matters, typically by simplifying the hierarchy and/or making more items static.

Symptom: Loading a screen, form, view, or data item takes too long

Possible causes	Solutions
The data required may not be arriving quickly enough, because on-line processes are taking too long.	See on-line processes, below.
The program may need to pre-load more items than it is doing, or in some way pre-prepare what it needs.	Add caching and/or preloading.
The program may need to anticipate your moves in some way, so that it can have the data ready.	Add an anticipatory prefetch task.

Symptom: Certain on-line processes take too long.

Possible causes	Solutions
There is an optimal split between the work-load of the client (terminal) program and the back-end (server) program. If either side is doing too much work, the task will run slowly.	Check the work-load split to see if it's correct. When using a data-base system, a good rule of thumb is that any request should issue only one SQL request to the server. If more are required, a stored procedure should be used.
Certain data needs to be prepared in advance, and it is not ready.	Prepare it overnight, or somehow before it's needed.
The process in question may be written poorly.	It may need to be rewritten.

Symptom: All on-line processes take too long.

Possible causes	Solutions
Although possible that there are problems which each process individually, the likelihood is that the system as a whole is running sluggishly. If there is more than one tier at the server-side, there may be too many tiers, and thus the work may be broken into too many parts.	If so, try to combine the tiers and the work.
There may not be enough tiers.	One server may be trying to do too much work. In this case, split the work up.
There may be too great a work-load coming in, and more processing nodes may be needed.	Get more.
There may be some overall processing issue, such as data base design, that needs to be corrected.	This may likely involve some expensive redesign and recoding.
There may be a faulty network connection which is causing packets to have to be resent or to wait.	Check and repair the hardware, if needed.
The server may have been compromised, and may be doing others' work.	Run a security check. If a problem is found, fix the hole first, then remove the offending program.

Symptom: Certain overnight processes take too long.

Possible causes	Solutions
Generally, this means that the slow task is poorly written.	Find the flaw and fix it.

Symptom: Overall overnight processing takes too long.

Possible causes	Solutions
This may be a general data base design problem.	If so, find and correct the data base flaw.
If two batch job-steps run at the same time, and they interfere with one another, both will slow down.	In this case, make one wait for the other to finish.
Two batch job-steps may not interfere with one another, yet be scheduled consecutively.	In this case, reschedule them so that they run at the same time.
If the batch server is not a secure mainframe, it may have been compromised.	Do a security sweep and correction, as above.

Symptom: The program lets you request some action, then issues an error message stating why the request can't be honored.

Possible causes	Solutions
To some, this doesn't sound like a problem. However, it is.	The program should not present the user with options that would later be rejected.

Integrity

The category of data integrity covers situations in which some or all of the data is incorrect, or becomes incorrect.

Symptom: One or more processes always gives the wrong result.

Possible causes	Solutions
One or more of the processes has flawed logic.	Find the flaw and correct it. When the flaw is found, look for similar occurrences of the same problem.
In a system that handles money or anything of value, there is always the remote possibility that somebody is stealing.	Find out what the cause is. Be aware that you may have to find out why the flaw is present, and what's really happening. Don't take this as paranoid. We've been on contracts where this was going on.
The same piece of data may be stored in two different places, and not updated quickly enough in one of the locations.	Rearrange the data usage so that the data is only kept once, in the place updated most quickly.

Symptom: One or more processes sometimes gives the wrong result.

Possible causes	Solutions
The same piece of data may be stored in two different places, and not updated quickly enough elsewhere.	As above.
Two users may be updating the same data at the same time. One's changes may be lost.	The programs must enforce record-locking, so that only one user can be modifying a given item at a time. The others will have to wait, or be locked out.
The program may have a clock-dependent flaw.	Of course, find and fix. Note that this is the hardest type of bug to fix.

Symptom: The program crashes.

Possible causes	Solutions
Bad code. Don't settle for this sort of thing.	Have this fixed first.

Symptom: Every time one thing is fixed, something else breaks.

Possible causes	Solutions
The complexity of the system is too much for the programmers.	There are a number of things that can be done here. Make sure that the program design is documented, in simple English. Make sure that the program is internally commented anywhere the internal code's functioning is not obvious. Make sure that the data base is commented too, using data base Comment commands. Make sure that the data base has every possible internal constraint. A constraint is a special type of coding that will make the server stop a program before it introduces bad data into the system.

History

Your system not only needs to know the status of your business today, but it must also let you look at the past and plan for the future.

Symptom: I can see data for today, but can't see what the status was yesterday, or what it will be tomorrow.

Possible causes	Solutions
The system is not keeping history. For instance, rather than an insured's start and stop date, it might have a status, of Covered or Not covered.	Remove the status flag and add start and stop dates. Then, you can see the status on any given day, including today.
The system is keeping history, but is also keeping status flags that are not updated at the right time.	Depending on the need for immediate data, either remove the status flag (usually the best option), or have the status flag updated at every driving event, including change of date.

Symptom: The system can't correct itself for changes. For instance, if it turns out that some billed-for event didn't occur, or occurred differently, changing that line item won't correct invoices, A/R, A/P, and G/L.

Possible causes	Solutions
Not enough history is being kept.	Keep track of the way things were last time a bill or other significant event happened. Compare what needs to be compared, and find and cope with the changes.

Sizing

Although disk space is relatively cheap, too much data can mean too much time, and too much program can mean too much money.

Symptom: The data is too big to store effectively or cheaply.

Possible causes	Solutions
Data base design flaw: Data is being stored redundantly.	Rearrange the data so that redundant data is not being stored.
Data base design flaw: The fields are not of the right type to store the data in a smaller form.	Redefine the fields.
The data itself is large but packable.	Pack some data.
Data is being kept that could just as easily be purged or bucketed.	Purge where possible. Bucket (collapse to a summary) where purging won't work.

Symptom: The programmers can't deal with the code-size. It's huge.

Possible causes	Solutions
The code is not written tightly.	Go through the code and clean it up.
Subroutines are being written and only used once.	Such routines can be included in the calling routines, and the subroutine eliminated.
There is code which is never called, usually because it's obsolete.	Find the number of callers for each item, and remove the unused ones.
Code may have been copied or cloned.	Two similar routines or programs can be combined by separating the common code off into a subroutine, and by having the other routines call the common subroutine. In a data base context, this can also mean a stored procedure or stored view. Two similar programs can also be combined into a single program that makes a decision to run one way or another, depending on the caller's intent.

Symptom: There are a number of similar systems, each with its own set of code and data. Each requires separate staff, or separate maintenance.

Possible causes	Solutions
Copying or cloning.	As above, see if these can be combined.

Networking

Systems in the enterprise should be able to coordinate with one another and share data.

Symptom: The systems won't talk to each other.

Possible causes	Solutions
The existing systems were designed separately.	Change the interface on one of the systems so that it will take to the other.
The existing systems were designed separately, and there is reason that neither should be modified.	Add a system running middleware, so that the data can be transferred without any modification to either existing system.

Security

This topic covers physical, software, and data security.

Symptom: People have access to data they shouldn't.

Possible causes	Solutions
The problem is physical security.	Add locks and other access control to segregate the users.
There is access to the data outside of the programs.	Install operating system-level access controls.
There is too much access within the program.	Have the program have two classes of user, and deny access where appropriate to the given data.

Symptom: People on the outside are getting my data.

Possible causes	Solutions
It may be that users have too much access to data through your application.	See above.
It may be that hackers have taken over your machine.	See below.
If you have an internet application, very often the program will send out a type of page called a "form". Hackers and well-intentioned power users can make changes to these forms, doing things you didn't intend.	Your application shouldn't trust a form to come back in acceptable shape. It must verify everything.
If *any* of your data or passwords goes over any part of the internet in an insecure, unencrypted manner, *all* of your data can be compromised.	Use and enforce tight security on all transmissions. Get control of what is going out over eMail, and use private networking or VPN for any point-to-point transmission leaving the building.

Symptom: Hackers are taking over my machine.

Possible causes	Solutions
Your operating system has a basic security flaw.	Fix it, patch it, or change systems.
Your people are allowing outside software in.	Don't allow people to bring in their own software. Don't allow the use of a browser that will run outside programs such as ActiveX components. Don't allow the use of an eMail program that will run incoming applications or macros.
Your server is running services you are unaware of and/or don't need to have running.	Stop all unneeded server services. If your operating system can't manage or control its service list, then you need to have a separate firewall do the job.

Symptom: You can't truthfully say that you can recover from a problem the next day, such as a fire.

Possible causes	Solutions
You need to be ready. Software may wipe out your system. Your hardware may fail today. The building might burn down tonight.	Have a complete set of back-ups, current and near-past, on-site and elsewhere. Have an arrangement with your vendor to deliver duplicate hardware on an emergency basis. Have an arrangement with your phone carrier to be able to transfer all of your calls elsewhere. Have an arrangement with a real-estate agent to be able to rent facilities on an emergency basis.

Work-flow

You got the computer to ease work-flow, remember? Let's get back to that position.

Symptom: We produce reports or spreadsheets that the office staff then uses to produce reports or data that we need.

Possible causes	Solutions
There are too many steps in the work-flow. The computer is supposed to produce final results only, not intermediate data for people to work with.	Analyze what the final product needs to be. Have the computer produce that, instead.

Symptom: A lot of paper is being produced. Can't this stuff be on-line?

Possible causes	Solutions
Your system may be producing completely unneeded reports.	In this case, drop the report entirely.

Possible causes	Solutions
Your system may be physically printing reports when other means of delivery are more appropriate.	Your system can produce its reports on-disk, as plain text, spread-sheets, web-pages, web-sites, RTF files (readable by many word-processing applications), Crystal Reports, and/or Adobe PDF documents. The system can place these in a known location, post them on-screen, eMail them to those who need to receive the report, or eMail a link to where the reports have been placed.

Reality

Sometimes you have to face the truth — but you need to find out what the truth is, first.

Symptom: You are told, "We can't do that."

Possible causes	Solutions
You are asking for something that can't be done.	Rethink what you want; scale back.
You are asking for something that shouldn't be done. For instance, forcing a balance in the GL.	Take your programmers' advice; come up with a more reasonable plan.
Your programmers don't know how to do what you want.	Find someone who does.

Price

Keeping costs down is important.

Symptom: Your solutions vendor costs too much.

Possible causes	Solutions
It's not the hourly rate of the vendor that counts; it's the price/performance ratio. A company that works three times as fast for twice the price is cheaper.	Evaluate vendors on that basis.
You change your mind. Each time you do, it's more time and more cost.	Design the system before you start. Often, this involves comp screen-shots, sample reports, or a manual.
You don't know what things will cost.	Try contracting on a fixed-price basis, once the specification is written.
You might be asking for more than you need.	Separate out your request into Need and Want. Go with need first, but let your programmers know what you would eventually like to have, so that they can plan things out.

Possible causes	Solutions
Changes over time: You have a base system, then add a feature, then add another…	Adding one feature at a time is fine, but there should be a road-map. If the programmers know what's coming, they can plan ahead for it, and not rewrite code.

Control

If you can't control the project, then you can't control cost or effectiveness.

Symptom: It's hard to control the programmers.

Possible causes	Solutions
Some programmers just won't follow instructions.	One "last warning" usually works, especially if the programmer knows that he was hired to replace the last programmer who didn't heed the last warning.
The program produced will only be as good as the programmers' understanding of the problem and the business process.	Make sure they have a complete understanding.
You should be able to meet with or talk with the programmers when you need to.	They should be close enough to do this. They should also have a good understanding of written and spoken English (and any other language the job requires).

Note that this is not an exhaustive list.
If you have a problem you need additional help with, give us a call.