Information technology is disrupting colleges and universities, forcing them to reevaluate their missions, methods, and business models. Everyone knows this. Even that bowtie-wearing emeritus professor who stubbornly clings to his IBM Selectric typewriter understands that the institution must adapt or die.
Of course, no one knows exactly what the colleges and universities of the future will look like. But we only need to consider other industries—like publishing, music, and retail shopping—to be reminded that we all—faculty, researchers, administrators, and technologists—urgently need to pull together to formulate a successful response to this changing landscape.

Still, if this is so important, and everyone knows it, why aren't IT departments, the home base in higher education for those who specialize in the application of technology, more central to the conversation?

**Notoriously Poor Relationships**
The answer seems to lie, at least in part, in the notoriously poor relationships between those in IT organizations and their consumers. As consultants to technology leaders in numerous higher education institutions and commercial enterprises, the two of us have noticed a recurring pattern of dysfunction in the relationships between IT people and those in the rest of the institution. There's a problem with the human interface.

This statement rarely surprises anyone on either side of the divide. Administrators are so used to projects being late and over budget that they don't trust developers or project managers at all. Developers are so used to administrators who constantly change their mind about what they want and when they want it that developers dismiss them as emotional and illogical. Students come to expect poor service from the support desk, so they turn to commercial technology and avoid using campus resources. Provosts give up on asking CIOs about the status of important initiatives because they're frustrated by detailed, jargon-filled answers that they don't understand.

The result is that technical and non-technical people often have disproportionately negative experiences working together. Geeks and non-geeks muddle through in a climate of mistrust and low expectations. And crucially, they are avoiding each other at the moment when they most need to collaborate.

**Collaboration and Connection**
The EDUCAUSE 2013 list of the top-ten IT issues facing higher education, as revealed in the article in this issue of EDUCAUSE Review, clearly demonstrates that technical leaders are fully aware of the transformations taking place in higher education. The prescient article subtitle—“Welcome to the Connected Age”—speaks directly to the opportunities and challenges of this moment.

Colleges and universities recognize the potential and the necessity of connecting people through technology. Nevertheless, little attention has been paid to the human connections required to create the enabling technological infrastructure. Developing, deploying, and supporting the technology that will fundamentally transform the campus experience cannot be done by geeks in isolation.

The four key strategic priorities identified in the article bring this situation into stark relief.

1. **Contain and reduce costs.**
   Without mutual trust between those in the IT organization and their consumers, will institutions be able to perform the miracle of integration across silo’d departments to minimize operational costs?

2. **Achieve demonstrable improvements in student outcomes.**
   Without effective cooperation among faculty, administrators, and IT staff, will higher education institutions discover new ways to measure, improve, and demonstrate student outcomes?

3. **Keep pace with innovations in e-learning and use it as a competitive advantage.**
   Without creative collaboration with the IT organization, will faculty be able to experiment with ways to transform the technologically mediated student experience?

4. **Meet student and faculty expectations of contemporary consumer technologies and communications.**
   Without an open dialogue with technology consumers about their expectations, will those in the IT organization be able to create optimal user experiences?

Clearly, if colleges and universities are to face head-on the challenges produced by the changing technology landscape, more collaboration between technical and non-technical people is needed. And if geeks and non-geeks are to improve their interaction, they need a much more helpful way of looking at why they find it so hard to work together and what they can do to improve.

**The Difficulty of Working Together**
Three years ago, we (the co-authors) set out to find an answer to the question of why technical and non-technical people have such difficulty working together, since the dysfunction between geeks and non-geeks was so obviously destructive and wasteful. But none of the common
answers we found were satisfying or compelling. The idea that technology people are simply a "different breed" doesn't really explain anything. The premise that non-technical people need to learn more about technology seems impractical and unrealistic. And although the two groups do “speak different languages,” that fails to account for the depth of the mutual mistrust and misunderstanding.

As we explored the problem, we noted that institutions had tried many things to improve the way technical and non-technical people work together, including radical process revisions, relationship management, and soft-skills training. But there was still no measurable uptick in the rates of project failure, no real diminution of frustration. And although IT people often had learned to “speak the language of business,” this had not resulted in any great unleashing of the productive potential of collaborative insight.

Eventually we realized that something deeper was at play. The troubles are caused by the fact that geeks and non-geeks see the world in fundamentally different ways. The mutual impatience, misjudgment, and mistrust are triggered by views defined by assumptions about how the world works and should work. As the two of us worked together, as a geek and non-geek, we felt firsthand, in painful detail, how corrosive these assumptions can be when left unexamined and how simple they are to accommodate once people become aware of them.

Different Worldviews

Different worldviews arise for two key reasons:

1. We choose careers based on our natural proclivities. IT people chose to work with technology because they feel a natural affinity for the predictability of code, the logic of math, and the joy of problem-solving. On the other hand, poetry professors were excited at a young age by the beautiful multilayered meanings of language that gracefully captures a deeply felt emotion. Whereas the poets among us bask in the delicious ambiguity of language, the programmers among us try diligently to root it out.

2. Our natural proclivities are reinforced by education, training, and isolation. The disciplines we choose reinforce our belief in the correctness of our own worldviews while blinding us to the existence of alternatives. The more the world with like-minded people, the more our preferences become axiomatic truths about life.

When working with others from our own disciplines, these assumptions about life and work serve us well. But when we work with colleagues who have incompatible assumptions, we get stuck in a cycle of constant misunderstanding and frustration. These assumptions even lead us to inadvertently violate each other’s deeply felt values, resulting in confusion, annoyance, and offense. And all of this havoc is wreaked by well-meaning people who often have no idea why this happens, because most of these assumptions are both unarticulated and unexamined. They are habitual beliefs, taken for granted. They are axioms. And when they conflict, watch out.

The dynamic of contrasting axioms was such a powerful frame for understanding dysfunction that we gave the idea its own word: contraxiom. A contraxiom is a matched pair of contrasting axioms that give rise to different worldviews. In all, we have identified seven distinct contraxioms between technical and non-technical people. Here we’ll look at the three that we see as the most common obstacles for information technology in higher education.

But before we explore these three contraxioms, please note that we are not advocating stereotyping. As we describe the differences between technical and non-technical people, we are setting up archetypes—fictional constructs that describe attitudes and behaviors occupying opposite ends of an imagined continuum. Each person is an individual and will reflect some combination of both archetypes. The purpose of archetypes is not to impose labels or judgments but, rather, to help reveal the patterns that appear in work relationships. We find that people are less judgmental of their colleagues once they recognize that their differences are emblematic of a different worldview.

Contraxiom #1: Work

For technical people, work is all about solving problems. They go to work each morning expecting to find, and solve, problems. If e-mail isn’t working, they can fix it. If a high-performance computing environment is needed, they can build one. In this worldview, everything is conceptualized as either a problem or a solution. It is no wonder that people with this mindset gravitate toward technology rather than literature. Non-technical people, on the other hand, tend to think that work is about achieving a vision. Of course, these two approaches could not be more different. A vision is not a problem; it is an imagined experience set in the future. People with this worldview go to the office each morning imagining a better future, and they work to create one.

This contraxiom often causes problems when people try to plan projects, because it leads to incompatible ap-
Crafting powerful problem statements is the single most important thing someone can do to motivate technical teams. A problem statement is like a mission statement, but it is more firmly rooted in the concrete reality of what needs to be fixed or changed. A problem statement is something that every geek instinctively responds to because it clarifies the murky ambiguity of everyday life, echoing the familiar structure of well-formed math problems. Most important, a problem statement tells geeks how they can win.

A problem statement does not necessarily describe a problem. It does describe something that can be solved. It sets up the possibility of arriving at a right answer. (Geeks find right answers very rewarding.) The problem statement might be about an issue, describing what’s wrong with the current situation. It might be about opportunities, or it might clarify objectives.

A good problem statement, one that really motivates, taps into what geeks find irresistible. Here are four ideas for what to emphasize in order to make a problem statement more motivating:

- **Value:** Geeks love to work on something that will make a difference, something that will provide a measurable and meaningful impact.
- **Difficulty:** Geeks love the challenge of problem-solving and are particularly excited by problems that may seem insurmountable at first.
- **Learning:** Geeks love to learn new things and expand the domain of their mastery.
- **Competition:** Geeks love the opportunity to prove that they deserve to be on top of the geek hierarchy. If others have tried and failed, the challenge is even sweeter.

An excellent problem statement, one that rallied the brightest technical minds of a generation and changed the course of human history, was delivered by U.S. President John F. Kennedy in May 1961: “I believe that this nation should commit itself to achieving the goal, before this decade is out, of landing a man on the Moon and returning him safely to the Earth. No single space project in this period will be more impressive to mankind, or more important for the long-range exploration of space; and none will be so difficult or expensive to accomplish.”

Problems. Since technical people think about work as problem-solving, they plan by carefully identifying the problems that will be addressed and methodically laying out the steps that will be followed to solve those problems. In short, they plan by working forward in time. Since problems exist only in the present, technical people focus on issues or opportunities rooted in the here-and-now. Non-technical people approach planning in exactly the opposite manner. Because they view work as achieving a vision, they plan by imagining a future state and plotting out the steps required to create it.

When these incompatible approaches collide, everyone feels uncomfortable and frustrated. Technicians wonder why their colleagues are such fuzzy, fantastical thinkers, too scattered to plan real work. On the other side, non-technical people feel that their technical colleagues are rigid obstructionists, incapable of understanding the big picture. Once people recognize that different, but perfectly understandable, assumptions are causing the confusion, everyone can let go of the frustration and judgment and make small accommodations to improve working relationships.

### What to Do about Problems vs. Vision

Both technical and non-technical people need to

- allow time to clarify what to do before worrying about how to do it;
- work together to write a problem statement that translates a vision into a specific achievable objective; and
- recognize that colleagues are not trying to prevent contributions when they shift the conversation toward either a vision or a problem.

### Contraxiom #2: Knowing

Technical people believe that something is true because it has withstood the rigors of detailed analysis. They don’t trust subjective feelings, even their own, as evidence. They analyze an idea to ensure its objective truth by breaking it down into its constituent parts and verifying the truth of each one. For them, details reveal truth. Non-technical people are more likely to trust an idea because it feels right or is confirmed by other people. They place more value on subjective experiences. They’re more likely to trust their intuition. In fact, the endless analytic details that give technical people confidence interfere with the more subjective verification process of non-technical people. For them, details obscure truth.

The problem is that we use the wrong tactics when trying to influence each other. We tend to use approaches that we ourselves find convincing but that leave our colleagues unmoved and unpersuaded. Imagine the following scenario. A university registrar is concerned that the upgrade to the student records system won’t be ready in time for the fall semester, and she expresses her concern to the project manager. The project manager confidently points to the monitoring dashboard, which shows...
**Persuasion: The Art of Adjustment**

Imagine that you want your university to create a mobile app that integrates with Facebook for tracking, searching, and notifying the community about student activities.

To build it, you need to persuade both the highly analytical CIO and the extremely effusive Dean of Students to go along with the idea. If you give them both the exact same pitch, you're probably not going to get the idea approved. To persuade them, you need to present information to each of them in their preferred style.

For the CIO, you should use an analytical structure in which you enumerate the key benefits and support them with facts, figures, and cogent logic. Make sure to reference how the benefits are achieved and to acknowledge any risk. For example: “Building this app will have four primary advantages. (1) Increase student participation. When students see what their friends are doing, they will be more likely to participate. (2) Increase long-term development. Our internal research shows that after graduation, students who participate in more than two activities per month donate $500 a year more than non-participating students. (3) Help optimize our investment in student activities by providing participation and inquiry data. (4) Expand our in-house mobile app development expertise on a relatively low-risk project.”

For the Dean of Students, you should use a more experiential, narrative format. Don't sidestep the benefits, but emphasize how each benefit feels. For example: “There are four main reasons to create this app: (1) Students will love broadcasting where they are and what they're doing on campus. And when they see what their friends are doing, they'll want to try new activities themselves. (2) Regina in the Development Office will love this too. Since students who participate become donors, she could be getting $500 more a year for every student who gets more involved. (3) We can start getting better data about what activities students really use. Right now, we don't really know who attends what. And if we have to tighten our belts, we can be sure that we don't cut something that students really love. (4) You must have heard that our cross-town rivals have a mobile registration app. If we don't do something, we'll start to look behind the times. This project is a low-risk way for the team to get the necessary skills for app development. And it will be great to have app development in-house, because it will be easier to come up with new ones in the future.”

exposing the “excessive optimism” of the status reports. She decides that the project manager is a mindless, robotic “process monkey” who is incapable of understanding the big picture. At the end of this conversation, both the registrar and the project manager have lowered their opinions of each other and have reduced the probability that this project will be completed successfully. They now lack the mutual trust required for an early identification of issues and intervention.

**What to Do about Analysis vs. Intuition**

Technical people need to

- use anecdotes, stories, and analogies to help their colleagues relate to their information;
- avoid responding to their colleagues' confusion by piling on additional details that interfere with understanding; and
- recognize that their colleagues' intuition often contains valuable insights based on personal experience.

Non-technical people need to

- support intuitive insights with evidence and logic and not expect other people to trust their gut instincts; and
- articulate the component parts of ideas and their logical progression, to give their colleagues confidence in the conclusions.

Had these approaches been employed in the scenario above, the conversation might not have ended in a dysfunctional impasse. The project manager and the registrar could easily have brought up their concerns using the language of the other. For example, after the project manager cited the current status of the project as evidence that there was nothing wrong, the registrar could have asked to review the upcoming tasks. Instead of implicitly rejecting his approach by ignoring past tasks and estimates, she could have discussed whether her concerns about the project
team would likely result in future tasks becoming late. On the other hand, the project manager, instead of implicitly rejecting her concerns about the team, could have shared with the registrar how he had overcome these types of problems on past projects and what he was doing to ensure the success of the current project.

**Contraxiom #3: Language**

When people talk about how difficult it is for technical and non-technical people to communicate, they usually say, “We don't speak the same language.” Of course, incompatible jargon is an impediment to communication, but the real problem is that we use language for fundamentally different purposes. Technical people use language primarily to transmit information, whereas non-technical people tend to use language to share meaning.

Geeks use language to describe the objectively verifiable world. They use it to encode facts and ideas that are stored on the hard drives of their own minds so that those ideas and facts can be transmitted and reproduced exactly in the minds of others. For them, good language is like code—precise, unambiguous, and complete. Non-geeks use language to synchronize their inner experience of the world with other people’s inner experience of the world. Their use of language includes facts and ideas, of course, but it doesn’t stop there. They also expect that when they speak, they are conveying their feelings, however subtle, about the facts. They expect that their listener will register the information and the meaning and will then respond with agreement or contrast.

If a non-geek says, “That chair is an unusual shade of red,” the statement might be followed by, “and it’s really beautiful in this light.” If a geek says, “That chair is an unusual shade of red,” the statement might be followed by, “and its RGB value is #DC143C.” This difference is subtle, but profound.

There are two common ways that this contraxiom undermines collaboration:

1. **We fail to share a sense of what is important.**
2. **We fail to share a sense of commitment.**

Because technical people focus on facts rather than feelings, they don't explicitly express what they consider important. For them, importance is determined by the logical extrapolation on the facts of the situation rather than a subjective judgment. They assume that if you share an understanding of the facts, then you must agree on what's important, so there's no reason to talk about

---

### The Difference between Commitments and Promises

One of the most common triggers of mutual mistrust occurs when geeks and non-geeks discuss deadlines. Technology consumers quite naturally want to know when they can expect to start using their new stuff. They need to plan for the conversion and communicate their plans based on the date. But geeks hate giving time estimates. This is not, as many non-geeks suspect, because they don't want to be held accountable for their work. The primary reason is that for geeks, estimates are lies. No one really knows when something will be done, and to geeks, lying is a grave sin, an assault on truth.

The way out of this problem is for both geeks and non-geeks to recognize the difference between commitments and promises. A promise is a guarantee of what will happen in the future, and as we all know, no one really knows with complete certainty what will happen in the future. A commitment, on the other hand, is an expression of what you care about and what you feel determined to do. The beauty of a commitment is that, unlike a promise, it does not guarantee an outcome. There is no risk of having told a lie if the result isn't achieved. So many things happen outside of our control that talking about the future with any degree of certainty makes many of us uncomfortable. A commitment guarantees only one thing: that you will continue to work toward an outcome. You can't control what happens, but you can control what you are committed to.

When non-technical people ask, “When will it be done?” geeks hear a demand for a promise that they can't honestly give, so they resist. Yet though some non-geeks may want a promise, most would be perfectly satisfied with an estimate and a personal commitment. Notice the difference between these two responses to the question:

1. I will finish the project by September 1.
2. I am committed to finishing the project by September 1.

For non-geeks, who use language to convey meaning, these two responses probably sound rather similar. Both express a personal commitment to finishing the project by that date. Non-geeks typically would not interpret the first statement as an ironclad promise, since they know that no one is a fortune-teller. But for geeks, who use language to transmit information, these statements are completely different. Few geeks would be comfortable offering the promise stated in the first response. In contrast, most geeks would be perfectly happy with the second response, especially if they had researched the issues and considered the date to be a realistic goal.

When geeks and non-geeks share the common language and understanding of commitments and promises, these discussions can become more productive and less painful.
it. In fact, they may even be concerned that pointing out what they consider important may be vaguely insulting to you, since it might imply that they consider you incapable of understanding the logical implication of the facts.

For non-technical people, importance is less a mathematical function of facts and is more a subjective feeling that something deserves respect and attention. They desire that the other people on a team share their feeling that something is important. So a non-technical person’s language is likely to have many more references to how they feel about something: “That’s great.” “I can’t wait to get started.” “We’ve got to do something about this right away.” “I’m really worried about this.” To a geek, these expressions might seem like filler—nonessential information.

This difference means that they miss each other’s cues. Non-geeks are looking for cues of agreement about the sense of importance they just expressed. It’s how they gauge commitment, and it’s how they know the team will be on the same page. They generally need to feel that their coworkers share an emotional drive to fulfill the same goals. Technical people, on the other hand, want to know that their colleagues are committed to solving the same problems and that they share a sense of urgency based on the importance of the problem rather than the emotions of the team.

The result is that everyone questions each other’s commitment to the success of their joint enterprise, sapping enthusiasm and encouraging resentment.

What to Do about Information vs. Meaning

Technical people need to

- explicitly point out what they consider important to the project, using simple descriptions such as “I really want to ...” “I’m excited about ...” “I’m committed to ...”; and
- describe the implications of the facts that they understand.

Non-technical people need to

- ask colleagues what they consider to be the most important goals, obstacles, and outcomes of their work.

Conclusion

As college and university communities continue to strive to adapt to the new world being ushered in by technology, they need to harness the knowledge, creativity, and insights of faculty, researchers, administrators, and technicians. But they need more than just goodwill and intelligence. To avoid the dysfunctional cycle of misunderstanding and mistrust and forge productive collaborations, the people involved need to understand and adapt to their contraxiom-driven conflicting worldviews.

© 2013 Paul Glen and Maria McManus

Paul Glen (paul@leadinggeeks.com), geek, is the founder of Leading Geeks (http://leadinggeeks.com/), an education and consulting firm dedicated to helping technical and non-technical people uncover untapped potential for productivity, creativity, and innovation. As the award-winning author of the book Leading Geeks, a long-time columnist for Computerworld, and a 25-year IT veteran, Paul helps concrete thinkers navigate the murky world of human relationships.

Maria McManus (maria@leadinggeeks.com), non-geek and the co-founder of Leading Geeks, draws on her experience as VP of Product at iVillage and Director of User Experience at Disney to help businesspeople unlock creativity and productivity when working with technical teams.