amazon.com Key Challenges

Ben Slivka (slivka@)
10/05/99 Version 0.3
09/21/99 Version 0.2
09/17/99 Version 0.1

1 Summary 1
2 Building a Platform 2
2.1 Profitability vs. Long-Term Investment 2
2.2 Merchant vs. Platform Tensions 3
3 Land-rush vs. Continuous Improvement 3
4 Vertical (store) vs. Horizontal (company-wide) Tensions 4
4.1 Marketing 4
4.2 IT prioritization 5
4.3 User Experience 5
5 Minimizing Inertia in our Software Systems 5
5.1 Componentization 6
5.2 Handling Errors 6
5.3 Testing Environments 6
5.4 Hardware/Operating Systems Choices 7
6 Metrics 7
7 Company Attitude 8
8 Recruiting and Growing People 8
8.1 Campus Recruiting 8
8.2 Growing Leaders 8
8.3 PowerLab 8
8.3.1 "Middle Integration" 9
8.3.2 PowerLab Notes 9
9 International 10
9.1 Engineering work 10
9.2 How International fits into marketing, bizdev, stores, etc. 11
10 Action Items 11
10.1 Regular Company-wide Strategy Communication 11
10.2 Regular Company-wide Planning Process 11
10.3 Determine and Communicate Realistic Profit Expectations 12
10.4 Regular Project Status Reports, Project Post Mortems 12
10.5 Increase Investment in Metrics 12
10.6 Reinforce Recruiting Efforts, Establish Leadership Program 13
10.7 Postpone Aggressive International Push Until IT Systems are Ready 13
11 Living the eCommerce Lifestyle at amazon.com 13
11.1 Dutch Auctions for Parking Spaces 13
11.2 Amazon.com Anywhere Payments Accepted at Cafeterias 13
11.3 Use amazon.com Systems for Internal Purchasing? 13

1 Summary

Amazon.com faces many challenges as it makes the transition from "Earth's biggest book store" to "the one place to find and discover anything you might want to buy online". While the totality of what we are trying to accomplish has never been done before, we can learn many lessons from companies and organizations that have preceded us.

During my recruitment to amazon.com and my first four weeks working at amazon.com I was fortunate to meet with many key people, both 1-on-1 and in larger meetings. I met 1-on-1 and/or shadowed Erich Ringewald, Gene Pope, Rick Dalzell, David Risher, Kim Rachmeler, Jeff Bezos, Joe Galli, John Witham,

Maryam Mohit, Anand, Rajaraman, John Vlastelica, Stan Shimizu, Rebecca Nelson, John Overdeck, Neil Roesman, Harrison Miller, Ashish Gupta, Charlie Bell, Dwayne Bowman, Joel Spiegel, Jeremy Eskenazi, Bob Vadnais, Carl Gish, Joy Covey, Leslie Kilgore, Nayeem Islam, Eric Broussard, Peter Cohen, Shel Kaphan, Bruce Jones, Stig Leschly, Alan Caplan, David Drischell, Jason Kilar, Jennifer Jacobi, Al Vermeulen, Mark Britto, Sarvesh Mahesh, Chris Payne, Doug Boake, Randy Puttick, and Jaleh Bisharat. Bill Allocca graciously allowed me to rumage around in some of the OBIDOS source code with him. I spent two days in reviews for Joe Galli (Business Development, Marketing, IT, Music Store, Tool Store, Cards, Fargo, Auctions) and a few reviews with Jeff Bezos (zBubbles, Auctions). I also read ~360 messages from the software@ alias, which gave me a few insights. I give full credit for any good ideas included here to the people I talked to, while any bad ideas are certainly my own.

The "land-rush" efforts amazon.com has pursued to date have provided us with dominant positions in a few areas and footholds in many other areas. We are in an enviable position. But, if we are to achieve our future vision, we need to slow down on new land-rush efforts, so that we provide an opportunity for management and teams throughout amazon.com (and especially IT) to build a bit more structure and process. An ongoing effort to modularize our systems and organizations will provide us with a stable platform from which to expand aggressively with quality going forward.

The key to our success will be in striking the balance between investing in ourselves and our systems, versus reacting quickly and decisively to market threats and opportunities. Investments in the former increase our speed and agility for the latter! Balance in all things.

The most important first steps for Jeff and Joe are to : 1) communicate a clear vision and strategy for the entire company for the next few years, and 2) start a fast, low-overhead planning effort to allow us to maximize the use of our scarce resources to attack the most important problems. There is no substitute for having everyone in the company rowing in the same direction!

The remainder of this document describes the key challenges I see, and provides a few suggestions for how to address these challenges.

2 Building a Platform

A key assumption I have is that, over time, Amazon.com will have two key revenue streams: 1) from our stores as a merchant, and 2) from service fees as we build a platform for 3rd party sellers. Joe Galli told me [on 8/16/99] that when we are a $200 billion revenues company, our revenue stream would be split 50/50 between stores and platform revenues. I think that is a most excellent goal!

When I think about existing platform companies, Microsoft, Intel, and Cisco come to mind. Going back in time, AT&T achieved that status in the early 1900s. A big difference between Amazon.com and these companies is that they were each profitable early on and were thus able to self-finance the development of their platform. By contrast, if you consider AOL as a platform company, it started c.a. 1985, went public in 1992, and restated all of its past results in c.a. 1997 to wipe out all of the profit it had claimed in the preceding years!

In order for Amazon.com to reach the economies of scale of necessary to generate high margins, I think we're much more likely to follow the AOL timeframes.

2.1 Profitability vs. Long-Term Investment

Amazon.com is not currently profitable. The rate of IT spend as a percentage of revenues is increasing – currently at 8% [per witham@], even though Galli/Covey want it at 2.5%. I think we need to continue investing in IT near our current level – a 2.5% rate will choke off the resources we need to achieve platform dominance. I believe it will be 5-7 years before our platform revenue stream starts to kick in.

ISSUE: How will we manage Wall Street expectations (for near-term profitability) as we invest at the level necessary to establish our platform?

2.2 Merchant vs. Platform Tensions

Just as we have tensions today between individual store goals and company-wide goals (see below), when we have a platform business the tensions between it and our stores have the potential to be extreme. The GM of Books will be naturally upset if we allow Barnes & Nobel and Borders to use our platform, as any competitive advantage that our platform gave the books GM will be wiped out. The GM of Books will not be happy with a "level playing field".

Microsoft faced similar tensions when it built up the MS-DOS and Windows (platform) businesses at the same time as it built up the Applications (store) businesses. Contrary to what you may have heard or read in the press, MS did not advantage its applications groups over competitors like Word Perfect, Lotus (1-2-3), or Ashton-Tate (dBase). MS Word and MS Excel on the Apple Macintosh were market leaders years before Word for Windows and Excel for Windows, and you can be sure that leadership was obtained by great vision and execution, as Apple had no motivation to help MS!

The ideas (if not the code) and teams behind these great Macintosh applications were moved successfully to the Windows versions, and the innovation of combining Word, Excel, and PowerPoint together to build Office was not matched by the leading – but separate – DOS application companies. It also helped that Bill insisted the Microsoft Applications team invest in Windows applications early, so MS was several years ahead on the learning curve, as compared with Word Perfect, Lotus, and Ashton-Tate. These competitors, fearing that investing in Windows would only aid Microsoft, chose to focus on IBM's OS/2 instead. This proved a fatal mistake, as OS/2 did not become a high-volume operating system.

The Microsoft Systems division (aka "Windows") had a Developer Relations Group (DRG) whose job was to evangelize ISVs (independent software vendors) to build Windows Applications. They met with all the big (and many small) developers, in areas as diverse as word processing, spread sheets, databases, development tools, CAD, chip design, graphics, process control, and finances. Even though other groups within Microsoft were developing competing products, the DRG people were scrupulous about not leaking information between ISVs and the competing MS groups. The Systems group held design reviews (under NDA) early and often to get feedback from ISVs on emerging platform services. These reviews served the dual purpose of 1) getting solid feedback from developers with a real-world perspective, and 2) accelerating acceptance and adoption of the Windows platform. Developer "mind share" was key!

Similarly, IT will need to create a "Platform" team with a Seller Relations Group. The Stores will have to build up their own engineering team (which could still live under the IT umbrella) to meet their specific needs, as the Platform team will be focused on keeping it's entire set of core customers (key 3rd party sellers + amazon.com stores) happy. The platform team will converge to a regular release cycle (probably quarterly, pegged off xmas dates). Meeting commitments will become very important, and the "land rushes" that amazon.com experience today will (I suspect) most often start in the Stores. The "land rush" technology thus developed with migrate over time into the Platform as more generalized services available to any Seller, just as technology migrated from Applications to Systems at Microsoft (OLE/COM being only the most obvious example). [Note: This is just one possible organization – there are several permutations which also might make sense. Ensuring that organizations are aligned with customers is an important principal, in any event.]

3 Land-rush vs. Continuous Improvement

Amazon.com has been very successful so far in chasing all sorts of emerging opportunities: customer reviews, Associates, 1-click℠ purchasing, new store categories, and auctions are examples. In most cases, there was little up-front planning for these efforts, and a lot of intuitive decision making and scurrying by everyone in the company was required to pull these off in the nick of time.

The company mobilizes it's best people from all areas to get something like Auctions built and launched in record time, then those great people move on to the next "land rush", and Auctions is left with a skeleton crew to soldier on.

This was a wonderful, totally appropriate way to operate when amazon.com was a smaller, less established company. And we will need to continue to pursue "land rush" opportunities as they arise. But, we need to adjust priorities such that we have sufficient resources to devote to our established services. If not, we risk falling behind competitors and, what is worse, not delivering the level of customer experience that is required for us to become "the world's most customer-centric company."

When we talk about the vision and the strategy and the roadmap for amazon.com, many people told me, "Jeff Bezos has it all in his head, I wish we could get him to write it down." In the absence of such a (obviously) living document, Jeff becomes a bottleneck for the company.

See "Action Items" on page 11 for solutions: regular vision strategy communication, a budget for land rush, and a regular planning process.

4 Vertical (store) vs. Horizontal (company-wide) Tensions

ben's axiom 1: An individual human being will most successfully achieve his objectives when he is given a sufficiently narrow focus that precisely matches his skills and experience and passion.
ben's corollary 1.1: If the objectives are not challenging enough, the individual will be bored and may underachieve.
ben's corollary 1.2: If the objectives are too broad, the individual will be unable to prioritize among them and will also underachieve.

The individual store general manager (for books, music, tools, etc.) is intensely focused (and rightly so) on attaining (and then retaining) the #1 retail position for her product category. Anything that detracts from that relentless focus, regardless of the source, is bad. For example, the Music GM reported in the Galli review the week of 9/7/99 that adding Auctions links on music detail pages was cost her $5.4 million in Q3 1999 revenue. That detracted from her goals, but presumably helped the company. Was this the right tradeoff? We don't know. We don't have any systems of measurement to answer this question.

Joe Galli expressed (I think) exactly the right philosophy for amazon.com on 9/9/99 at the Business Development review: "We're going to err on the side of autonomy for the [store] GM, it is the only way he can beat focused competitors like eToys. Jeff and I don't have the time to match-up against the Toby Lenk [eToys CEO]."

And yet, if amazon.com is to be more than just a holding company for online stores, we also need to work on building common infrastructure, services, marketing, etc. that the individual stores can leverage.

This tension between narrow customer-segment focus and organization-wide integration is a classic challenge. We need to be explicit about these tensions, so that individuals and leaders in our various groups can manage them explicitly. Otherwise, these tensions will drive our people toward political behavior as they compete for scarce resources (primarily people, money, and executive attention).

These tensions affect every group/function in the company, but I call out a few specifically below.

4.1 Marketing

Amazon.com is still best-known as a bookseller. The brand needs to evolve to a brand promise of "the one place to find and discover anything you might want to buy [online]".

Some companies minimize this tension altogether by emphasizing the supremacy of individual brands (Proctor & Gamble being the classic example) and submerging the corporate brand.

ISSUE: Microsoft evolved to have a central marketing group that managed the Microsoft brand, and individual marketing groups to do targeted marketing for individual products or groups of products. I believe Amazon.com will need a similar structure – a corporate group to manage the brand, and individual marketing teams in each store (or group of stores) to do very focused marketing.

4.2 IT prioritization

How should IT divide up its resources between projects that are applicable company-wide, versus more narrowly targeted efforts which benefit only a single store? If all resources were devoted to company-wide efforts, individual stores would not be able to respond quickly to their competition. If all the resources were devoted to servicing individual stores, there would be an increased amount of duplication of code, and it is likely that the user experience would suffer as similar features (in the customer's eyes) would work differently in different parts of amazon.com.

ISSUE: At present, the process for identifying and prioritizing projects, and then communicating that prioritization, is very rudimentary. It is not clear who (other than Jeff@) is the final arbiter of the priorities. We need a just slightly more rigorous process to ensure that we capture all the potential projects, prioritize them with input from all partners, select the most impactful short-term and long-term projects, staff them appropriately, pick and commit to hard dates for specs, beta test, and launch, and then execute with precision.

4.3 User Experience

As above, there is a tension between making a truly great user experience for the Music store (for example) that is uniquely suited to Music customers, versus building a common shopping experience for amazon.com customers that works consistently no matter how you discover the product you want to purchase.

ISSUE: Who owns this? Maryam's Program Manager's think they do. But Jaleh said something in her Marketing review with Galli that Marketing should own the overall customer experience. At Microsoft, the product teams (program management and development, chiefly) drove the product definition and vision and execution, with input from customers, influential end users, marketing, product support, and executives. The primary responsibility of the marketing team was to communicate the benefits of the product in a focused way and execute amazingly well on all the other classic marketing tasks.

5 Minimizing Inertia in our Software Systems

As amazon.com grows – more stores, more ASINs, more technical people, more operations people, more marketing people, more source code, more data, more processes and procedures – it is inevitable that inertia will set in. In this section I focus primarily on the IT software development teams, but it is possible that these comments could apply to other groups as well.

To state the obvious, software development is very difficult. It was difficult in the mid-1960s when IBM developed the OS/360 operating system, about which Frederick P. Brooks, Jr. wrote "The Mythical Man Month". And it is difficult today, when Microsoft is struggling to complete Windows 2000 with an estimated 50 million lines of code and 3000+ developers, testers, and program managers.

The Internet does not make software development any easier. If anything, it is more difficult, since we must keep our existing web site and back-end systems running while we improve them. Packaged software (like Microsoft Windows and Microsoft Office), on the other hand, have the luxury of long product cycles and a "setup" process that renders the machine unusable while the software (and, in some cases, data) are upgraded.

Looking at just a few snippets of code, and talking to many developers here, leads me to conclude that our current source code base is no worse (and no better) than at many other companies with efforts of our scale. I'm most familiar with various code bases at Microsoft (MS-DOS, Windows 9x, Windows NT, Internet Explorer, and Hotmail), but also have some insight into Cisco and a few other code bases.

If we are to achieve our scale and platform goals, we'll need to invest in improving the structure of our code base and the tools we use to write code, test it, and deploy it.

5.1 Componentization

Today there are roughly 250 software design engineers in IT. Over 100 of those programmers (SDEs) work on Obidos, which is the program that generates the **entire** web UI for www.amazon.com. This program is built as a single executable file, so any time one of those 100+ developers makes a change, it has the potential to impact any of the other 100+ developers.

By contrast, when I led the Microsoft Internet Explorer 3.0 team, I had 34 developers working on several different components that were built independent of one another. The developer working on HTML rendering had little opportunity to negatively impact the developers working on the script hosting component, or the HTTP cache component, or the PICS rating component.

We need to factor our monolithic code base into relatively isolated components, so that our developers can innovate more quickly without distracting one another. The Enterprise Services effort under Peter Cohen is a step down this path, but we have a long ways to go. The leadership of IT all agree we need to do this componentization, but the precise order of attack is not clear. We simply need to increase our focus on this, which will mean doing fewer land-rush activities.

Factoring our software code base is intimately related to how we organize our teams. It used to be said that you could open a DEC mainframe or minicomputer and discern the organization chart within DEC by the layout of the hardware components. The same is true for a software organization and its dual, the software it produces.

There is no perfect organization – for people or for code. All we can do is identify the centers of gravity in our code (both as it stands today and as we think it might exist in the future), and then look at the leaders we have and make a reasonable compromise. Trying to jam people into a code-centric organization, or trying to jam code into a people-centric organization, doesn't work [insert your favorite Microsoft war story hear.]

5.2 Handling Errors

All good developers I know spend 80% of their design/coding/debugging time figuring out (and worrying about) how to deal with errors, exception cases, and anomalies. Given our goal of becoming "the most customer-centric company in the world", the quality bar for our software systems is going to get higher all the time. Reliability, Availability, and Serviceability (IBM's favorite "RAS") are key.

My biggest frustration as a programmer was not knowing what to do when one of my functions gets an error – does the function try to recover from the error on its own, or push it up to the calling function, or ignore it? In many cases, it's not clear what to do – sending an error back to the caller often doesn't help because the caller lacks any means to recover/retry the operation. But ignoring the error might mask a serious problem. As for trying to recover in the function itself, we encounter a recursive problem, as the function probably didn't get enough information to determine how to act!

As we improve our development methodology, our tools, our standard libraries, we need to be very thoughtful and pragmatic about how to make error handling much more concrete and effective.

5.3 Testing Environments

I am most appalled by the testing practices here at amazon.com. But, given how complicated it is to replicate the production execution environment, I am not surprised. Testing massively scaled, live systems like amazon.com, hotmail.com, and yahoo.com is a major challenge. I don't think any company has a great solution today. I do think it is worth a serious investment in IQ and people to develop world-class testing abilities at amazon.com.

Conceptually, in addition to our production system, we would have an exact clone of the production system to be used for "integration" testing (when several developers bring their code together to test that it all

works together before deploying to the production system), and each developer would have access to his own clone of the production system.

We should have "component tests" for each component that would run through all the functionality of the component in a pretty thorough way. We should have "build verification tests" (BVTs) for developers to run that automatically ran a set of components or the entire system through a series of standard quick tests. BVTs are a "sniff" test to make sure that nothing obvious has been broken. We should have "stress tests" that exercised individual components and the system as a whole at peak loads.

Each developer should have at their disposal, round the clock, a complete clone of the production system, so that he can component tests and BVTs prior to checking in his changes. And, in the case of major changes, he would be able to run stress tests, too. A separate testing team should regularly run component and stress tests on the system to catch problems that may have slipped by developers.

The expense in hardware, operations staff, and replication time precludes us from maintaining the 10 or 20 complete testing clones of the production system that would be very nice to have. Just getting all the data to replicate would likely bring the production system to its knees. We need to figure out some more clever solution, which would most likely involve putting hooks in our production system to support in situ testing. This is a computer science research project, but has big implications for us.

As a side note, at Microsoft, the ratio of developers to testers commonly ranged between 4:1 and 2:1 in the Windows and Office groups. In our IT group, by contrast, we're more like 20:1! That said, in the early days of Microsoft there were only developers. The specialization into program managers and software test engineers happened when MS was close to 10 years old.

5.4 Hardware/Operating Systems Choices

Today, 80% of our server budget is spent on Compaq (nee DEC) Alpha servers. The rest are Sun boxes and some NT machines. DEC Alpha is not a volume market leader, and so we can expect over time that this platform will not keep pace with the Sun and Intel platforms – advances in CPU speeds, I/O architectures, development tools, and packaged software will all likely trail the volume leaders.

While we probably don't need to do anything immediately, but over the next 6 months we should put together a plan to isolate DEC-specific usage in all of our systems, so that we are prepared to move to the best high-volume platform in a few years. If we don't, we risk ending up like Compuserve, which built it's original system on DEC 10 mainframes, and didn't shift off until it was much too late. Compuserve in the mid-1990s was having an outside firm custom-build DEC 10 clones (a 36-bit word, etc.) that cost at least 20x the price of comparable Intel compute power. And of course they had archaic development tools, etc. Compuserve finally started shifting to Windows NT in the mid-1990s, but it was too late, and AOL acquired them for their customer base and the Compuserve brand. This lack of investment in advancing their platform was a symptom of the broader problem that H&R Block viewed Compuserve as a cash cow, and so scrimped on a variety of necessary investments.

We should review our hardware/OS platform mix every six months.

6 Metrics

We have a lot of data about our web site, but there is a lot of information we don't have in order to run our business successfully. Here are a few examples:

1. **Customer NPV:** Today we compare various methods of customer acquisition based on Cost per New Customer (CNC). But the right measure is Customer Net Present Value – we need to know how much net profit we make on each customer. With that data, we can better prioritize our activities toward maximizing both the number of customers and our overall profits. I asked about this number at one of the Galli reviews the week of 9/7, and someone in finance promised to get back to me with some figures. I haven't heard a peep. ☹
2. **A complete store P&L:** Several of the store GMs commented to me that the don't have enough data to really run their Profit & Loss center. In particular, they have little visibility into the supply chain for

their products, and so have very little control over order, stocking, inventory, etc. that directly effect their margins.

You can waste a lot of time gathering, computing, and reviewing numbers. But if we can focus on a relatively small collection of the right metrics, we can improve upon the customer experience and increase our revenues and margins.

7 Company Attitude

A few people (more recent arrivals to amazon.com) have used the term "arrogance" to describe some aspects of our behavior. The examples cited are in recruiting and in our approach to business opportunities.

Coming from a world-class arrogant company (Microsoft), I have not noticed much of this myself. But it is important to remain vigilant here, as arrogance and hubris are very destructive to the long-term viability of our company.

We should be self-confident and positive in our dealings with others, be they customers or candidates for employment or competitors or the press. But we should do our best to avoid crossing the line into arrogance.

The tone for this is set at the top, and here we have an advantage over Microsoft, as Jeff Bezos is more polite and less arrogant than Bill Gates.

8 Recruiting and Growing People

8.1 Campus Recruiting

Establishing a campus recruiting "machine" for undergraduate and graduate students is critical. The Fall 1999 effort is going to achieve modest results because of our late start, and the CollegeHire deal will have mixed results.

Jeremy Eskenazi (the newly-hired head of campus recruiting) has fabulous experience here (from Universal and elsewhere), so I'm very optimistic that we'll have a great machine in place for Fall 2000 recruiting. As strong as the Microsoft campus-recruiting machine was, I believe Jeremy's will be even stronger!

A key element of campus recruiting is getting managers throughout amazon.com to drive elements of the effort. Jeremy will identify school owners, and hiring managers will be very active in recruiting trips to campuses and following up with candidates. To the extent managers within amazon.com are not used to this, we'll have to educate them about the value of campus recruiting and help get them into the swing of things.

8.2 Growing Leaders

This is a very important element of building a long-term company. For all of the great things that Microsoft did in my 14 years there, this was the key area where Microsoft fell down. The passion and focus for this has to come from the top, and Bill Gates does not have it. Jeff and Joe are going to have to drive this, and be role models for the rest of the organization, just as Jack Welch is at GE.

Over the next 6-12 months, we should start the Amazon.com Leadership Program, which will consists of classes, retreats, reading material, coaching, and mentoring. GE is probably a good role model here, at least as a starting point.

8.3 PowerLab

I attended a 5 day "experience" called PowerLab on Cape Code with 13 other senior Microsoft people in September, 1997. PowerLab has been run for almost 30 years by Barry Oshry, and during that time he has developed a model of how power operates in systems of people. His book, Seeing Systems : Unlocking the

Mysteries of Organizational Life, describes a model of "tops", "middles", "bottoms", and "customers". Coming back from PowerLab, I found that this model very accurately characterized the structure at Microsoft, and explained many of the problems that I, and other senior MS people, were seeing as Microsoft grew larger. I organized the Middle Management Retreat for 45 senior MS "middles" in November, 1997, where we spent a day learning this model, and a day (as a group decision) designing "Microsoft Version 3.0".

8.3.1 "Middle Integration"

Oshry emphasizes "Middle Integration" as a key behavior in healthy organizations. He defines this as middles working with other middles to help reinforce and achieve each others goals. As a rule of thumb, I believe the most direct way to ensure middle integration is for each group to **devote 10% of its resources to help other groups**.

As the company gets larger, and we struggle to find efficiencies, groups will become increasingly dependent upon one another. A common tendency for driven, passionate, successful people is to avoid dependencies, since that adds more risk to their project.

Instead, we need to imbue the culture with the philosophy that every manager should actively commit his people and time to furthering the goals of other groups. Of course the primary focus of each manager is to achieve his own goals. But enlightened self-interest will motivate helping other groups, as the manager can expect that by helping other groups, other groups will help him in return.

8.3.2 PowerLab Notes

PowerLab reinforced for me the importance of communicating clear goals throughout an organization, getting the right people in the right jobs, coaching, and playing the customer advocate role. I've included my very brief notes below. My apologies in advance for their rather succinct nature!

8.3.2.1 When "Stuff Happens to You

"The Side Show" – negative/typical response	"The Center Ring" – balanced response
Make up a story to explain what happened, and to justify my response	Be strategic – take their worlds into account
Take it personally	Try to understand, use empathy
React – get mad, get even, withdraw	Don't get hooked on stuff (emotions)
Lose focus	Stay focused

8.3.2.2 Patterns in Top/Middle/Bottom/Customer Systems

Door A:

Predictable Conditions	Predictable Responses	Disempowered Experiences
Top Overload	Suck it Up	Burdened (w/the situation)
Bottom Disregard	Hold "them" responsible	Oppressed (by "them")
Middle Crunch	Slide into the middle	Torn (by job)
Customer Neglect	Hold "it" responsible	Screwed (by "it")

Everyone blames something or someone else, instead of looking for a change in himself or the system.

Door B:

Take a stand.

- Be a Top who creates responsibility throughout the system.
- Be a Bottom who takes responsibility for your condition and for the condition of the whole thing.
- Be a Middle who stays out of the middle –maintain your independence of thought and action.
- Be a Customer who gets in the middle of the delivery process and helps them work for you.

8.3.2.3 Middle Empowerment Strategies

- Be top when you can – the leader
- Be bottom when you should – the reality check
- Be coach
- Be facilitator
- Be integrator

8.3.2.4 Customer Empowerment Strategies

- Find out how "it" (the system) works
- Set clear demands and standards
- Get into the process early as a partner, not late as a judge
- Stay close to the producer
- Develop relationships

8.3.2.5 Top Empowerment Strategies

- Share high-quality information
- Involve others in big decisions
- Ask for help
- Create structures
- Invest in training and development
- Create powerful visions
- Invest in relationships
- Create and use teams

8.3.2.6 Bottom Empowerment Strategies

- Make the organization's vision your vision
- Let tops and middles know what you need
- Look for opportunities to take action
- Hold other bottoms responsible
- Don't play the "bitching game"

9 International

Microsoft got aggressive about International markets fairly early in its lifetime. Bill Gates did a JV with a company/guy in Japan c.a. 1979, four years after founding MSFT. Scott Oki opened the first few international subsidiaries (for sales and support) in the early 1980s.. Chris Smith was the master of this, and opened I think 25-30 subsidiaries over a 5-7 year period, using a very rigid formula relating sales in country to allowed head count.

The MSFT engineering teams developed coding disciplines, standard code libraries, and localization tools that made it fairly easy for English-speaking developers to build single binaries that worked world-wide. The actual localization work (MSFT has 22 tier one languages) is carried out by teams in the US, Europe, and Asia.

For amazon.com, I think our systems are not sufficiently modular/distributed for us to aggressively expand internationally at present. I think we should go at this very slowly, even if local competitors emerge, until we have the proven ability to run multiple, geographically distributed data centers in the U.S.

9.1 Engineering work

Building software for local use anywhere is a fairly well-understood problem. You choose between Unicode (16-bit) character sets, or use a combination of code pages for single-byte languages and double-byte character sets for (typically) Asian languages. The sane choice is Unicode, which simplifies

programming at the expense of requiring twice the disk space and RAM and bandwidth (vs. ASCII). You can use data compression to minimize this impact in some cases.

Language strings no longer appear in any source code, so that you have a single, world-wide binary. Localized text is probably stored in a database (but cached in some in-Memory system to ensure that page formatting times are not slowed down). We'll need a more sophisticated HTML template system, since we'll have Right-to-Left languages (like Hebrew) and Top-to-Bottom languages (like Japanese).

And then we have the whole issue of distributed data centers: replication, faul-tolerance, data coherence, privacy, security, etc.

9.2 How International fits into marketing, bizdev, stores, etc.

There are a ton of issues here. Is there a GM per country per store? Most certainly not when we first enter a country. But over time, as our revenue base grows? How do you handle the local cultural differences for marketing and merchandising? What are the payback cycles – it seems like we'll need a pretty large team in the local country at first launch of a local store. MS, by contrast, could start a subsidiary with one person.

This is a big issue outside the scope of this document!

10 Action Items

10.1 Regular Company-wide Strategy Communication

Historically, amazon.com has had "dark" projects (like Auctions, zBubbles) that go on in secret for months and then suddenly show up on the radar screen of the rest of the people in Stores/IT/Marketing a month or two before launch.

While this approach has the potential benefit of minimizing information leaks that would tip off competitors, overall it hurts the company. As amazon.com grows, we need to continue to push down responsibility and authority to lower levels of people in the organization. For these people to be successful, they need to understand as much as possible about how their work fits into the broader context of amazon.com. In the absence of this understanding, conflicts will arise and senior management will be placed in the unenviable position of resolving these conflicts. Like coding defects, it's much better to avoid these conflicts in the first place.

A key element of our Strategy communication must be the identification of explicit tensions in our organization: long-term investment vs. profitability, land rush vs. continues improvement, platform vs. stores

ACTION: Jeff/Joe need to communicate quarterly to the entire company, reinforcing the long-term vision of the company and explaining the near-term strategic initiatives and priorities. First such communication should be published no later than 10/31/99.

10.2 Regular Company-wide Planning Process

Amazon.com has reached the size (people, number of projects, revenues) where we need to be just a little bit more rigorous in our planning process. The goal is to ensure that we are balancing the allocation of our scarce resources (especially people) among four key areas:

1. Strategic initiatives (e.g., "marketplace", planned new stores, etc.)
2. Continuous improvement of current service offerings (Books, Music, Auctions, etc.)
3. Development infrastructure (e.g. Enterprise Services, componentization, development tools, testing environments, etc.)
4. Opportunistic initiatives (e.g., "land-rush", competitive responses, etc.)

The process is pretty straightforward. Senior management will communicate 3-5 key priorities for the year, and **establish a tentative budget (% of headcount)** for each of these four areas. Business owners (primarily store GMs) will map out their needs and priorities for the next year, and take a swag at the range of possible economic upside for each feature request. IT, Program Management, DC, CS, FIN [any others?] will take these requirements, identify areas of overlap, coalesce the requests to produce "projects", and provide cost estimates for each project. Representatives from all parts of the company will go off-site to prioritize the list of projects, based on the company-wide key priorities, budgets, and costs, and there will be a final few weeks to tune and optimize the specs and priorities and costs, and then the plan will be published.

With this plan in hand, each team will be better able to build and manage dependencies between itself and other teams.

ACTION: Joe needs to kick of a planning process by 10/31/99, so the company is set to rock 'n roll as the holiday shopping season ends and Q1 starts.

10.3 Determine and Communicate Realistic Profit Expectations

We need to honestly assess how long it will take us to reach profitability without unduly restricting our investments in building a platform. If we can find a path toward building our platform that brings profitability more quickly, super! But we need to avoid being to short-term in our thinking, or will miss the big platform payoff.

ACTION: Jeff/Joe need to nail this down (with help, of course!) by 1/1/00.

10.4 Regular Project Status Reports, Project Post Mortems

There should be regular status reports (at least monthly) from all major project efforts, available to all technical staff. These reports will use a simple, standard format for reporting progress on milestones, highlights, lowlights, dependencies, and issues.

When a project is completed, the Post Mortem report is an opportunity to record high points, low points, lessons learned, and suggestions for the future.

These reports will provide a variety of benefits:

a) Customers will be kept abreast of the status of the projects they are dependent upon.
b) Individual project owners will collect enough data on schedule slips and cost overruns to improve their estimating abilities.
c) Archives of project status reports and the post mortem provide a wealth of valuable information for people new to amazon.com or new to the individual project to learn from.

ACTION: Dalzell needs to set up a framework and insist on delivery of reports starting 10/31/99. Risher should do same for his team. Up to Joe when other teams start doing this.

10.5 Increase Investment in Metrics

Joe Galli mentioned "our 20 key numbers" in a review the week of 9/7/99. We need to identify what those metrics are and then become a well-oiled machine at reporting and tracking them. We also need to reduce the inertia in our Data Warehouse and Decision Support Systems to enable business leaders to quickly get the most important data for them.

ACTION: Joe needs to identify the key numbers (via a team, process, etc.) and publish those by 11/01/99.

10.6 Reinforce Recruiting Efforts, Establish Leadership Program

I think we're making good headway on recruiting. Jeremy Eskenazi is a great hire, and key folks like Jeff Holden in IT are driving the campus recruiting effort this year. We're also starting a co-op relationship with University of Waterloo (in the early 1990s, 10% of all software developers at MS were Waterloo graduates).

ACTION: In the next 6-12 months, Joe needs to put a super-strong person in charge of leadership development.

10.7 Postpone Aggressive International Push Until IT Systems are Ready

I understand the land-rush opportunities, but as above I think we'll "break the camel's back" if we try to add these additional dependencies onto the IT group in 2000.

ACTION: As part of the planning process, Jeff and Joe need to permit the leadership team to prioritize international expansion against new stores, internal structural investments, etc.

11 Living the eCommerce Lifestyle at amazon.com

Just some nutty ideas I have for us to "eat our own dogfood".

11.1 Dutch Auctions for Parking Spaces

If we really think that alternate pricing and payment systems will become common place, let's start using them ourselves. PacMed Lot 1 has a fixed price ($62.50/months w/amazon.com subsidy), and a waiting list of 200 some people (also, yield management doesn't seem to be tuned, as there are always lots of free spaces). Why not hold a Dutch auction every month for one-third of the spaces? [For purposes of yield management, you'll issue more passes than parking spaces.] We would learn first hand how socially acceptable this pricing mechanism is, with the side effect of getting a market price for the parking spaces.

11.2 Amazon.com Anywhere Payments Accepted at Cafeterias

Eventually we want amazon.com payment systems accepted world wide, both on-line and IRL (in real life). Let's eat our own dog food here and have the cash registers accept payment from Amazon.com Anywhere-enabled PDAs (personal digital assistants). Again, the motivation is not immediate cost savings, but rather to experience first hand how these advances work in the real world.

11.3 Use amazon.com Systems for Internal Purchasing?

This may be too wacky, or it may just be too early to pursue. Or maybe it is too different from the consumer and seller systems we're building today.