IV. Proprietary Capability, Technology, and Content

- Products, Services, and Content
- Success Factors in Service
- Success Factors in Content
- Project Estimation and Management
- Prototyping and Software Engineering
- Software Specifications
- Software User Interfaces

Proprietary Capability, Technology, and Content (con't)

- Testing and Quality Assurance
- Product Enhancements
- Manufacturing
- Customer Support
- Training
- Intellectual Property Protection for Software

Services, Products, and Content

- ***Consider carefully the advantages and disadvantages of selling a service, packaging and marketing a product, assembling and enabling access to content, or various combinations of product, service, and content. (#21)***

Services, Products, and Content (cont'd)

- Products
  - High gross margins and growth potential, but...
  - High start-up costs, but very risky and competitive
- Services
  - Low start-up costs, and usually less risky, but...
  - Slower growth, lower profit potential
- Content
  - Slow growth or very high start-up costs
  - 1999 darling of investment community (portals), crash in 2000

Combinations of products, services, and content

- Service in products
  - Consulting, support, training
  - Source of differentiation and extra revenue
- Products in service
  - Packaged toolkits, methodologies, code libraries — Source of competitive edge
  - Packaged training videos — Source of expanded market, extra revenue
- Content in products
  - Templates, databases sold with software tools

Combinations of products, services, and content (cont'd)

- Leveraging, bootstrapping a products company from a services company
  - Cognos — Successful
  - HCR — Unsuccessful
- From service to products to products and service: Netron
- Products with content being the profitable "annuity": TaxPrep

Software Service Success Factors

- ***Clients contract for consulting and software-related services because they do not want to hire permanently or cannot find critically needed skills; your success as a consultant/contractor is based on the degree to which you can professionally, skillfully, and consistently deliver high quality and knowledgeable service such as advice, management assistance, software development, or training. (#22)***

Why clients hire consultants, contractors

- Unique distinctive competence
- Training, bootstrapping one's own staff
- Inability to find or hire qualified staff
- Lead time
- Another point of view
- Arbitrate internal organizational conflict
- Cover you’re a--

Consulting, contracting distinctive competences

- Knowledge of a particular application area
- Knowledge of a particular technology: foecasting, introduction, usage, optimization
- Ownership of proprietary technology
- Skill in computer systems integration
- Skill in project management of large jobs
- Skill in imparting knowledge to others
- “Either they want the service you have, or they don’t, and you change it.” (Karen Holtzblatt)

Marketing of consulting and contracting services

- The goal: To be known as the expert, e.g., James Martin, InContext Enterprises
- Professional achievements
- Industry involvement
- Articles in trade publications
- Books, newsletters, or industry reports
- References and “word of mouth”
- “How much do you know about your competition? Everything!!” (Josef Kates)

Success Factors in Content

- Speed to market
- Comprehensive and authoritative content
- Marketing budget
- Strategic alliances
- Revenue model (e.g., advertising, e-commerce, subscriptions)

Software Project Estimation and Management

- ***Software development, whether carried out for clients or internally for purposes of product development, is notoriously prone to cost overruns. You must therefore estimate carefully, monitor progress, and re-estimate assiduously, using methods such as “divide and conquer,” Delphi, and post mortem. (#23)***

Class project management exercise

- Recall a major success or failure in terms of delivering software on time and on budget with which you were personally involved or with which you are familiar.
- What were the top two or three reasons for the success or failure of this project?
- Answer briefly including with your full name

Estimating software projects

- Why it's hard
  - It's never the same
  - Specifications are vague, even for products
  - “Clients” change their mind, even internal clients, i.e., product managers
  - Communication problems with the client
  - Communication problems within the team

Estimating software projects (cont’d)

- Estimating
  - Divide and conquer, estimate smaller pieces
  - Delphi, combining independent estimates
  - Post mortem, comparison to past projects
- Regular tracking and re-estimation

Managing software projects

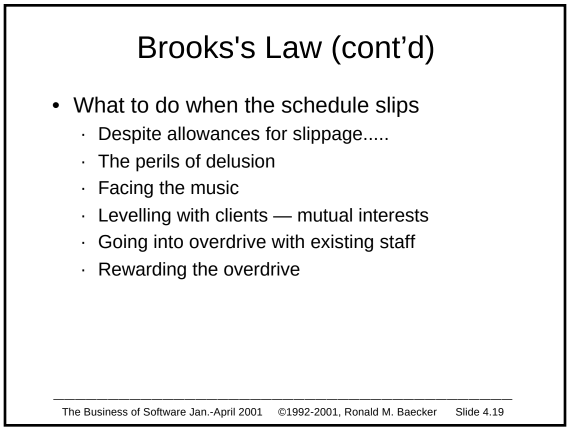
- Methodologies and management techniques
  - Thorough development plans
  - Design and code reviews
  - Regular progress reports (no excuses!!!)
  - Microsoft “synch-and-stabilize” approach
  - Open communication
  - Responsibility and accountability, hierarchical negotiated estimation

Managing software projects (cont’d)

- Tools
  - Project management systems
  - Workstations, programming environments
  - Tools for testing and documentation
  - Electronic mail, internal communications

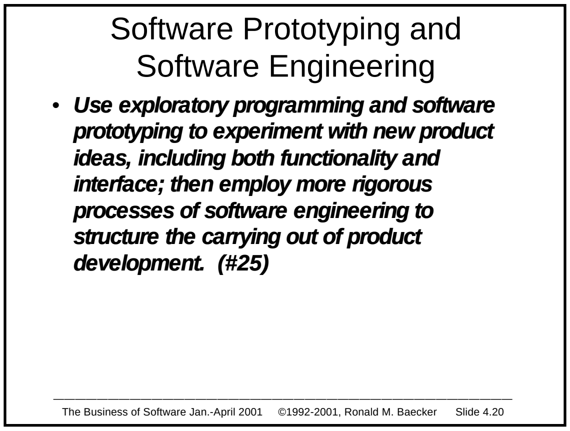
Brooks's Law

- ***Always remember Brooks's Law: Adding more manpower to a late project makes it later. (#24)***
- Why Brooks's Law
  - Start-up, training time
  - Communications time
  - Work reorganization time and impact



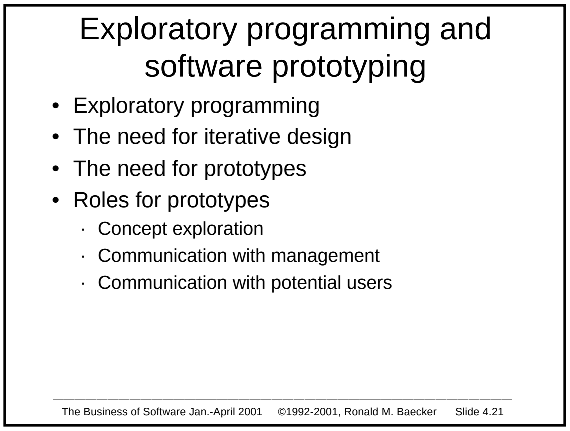
Brooks's Law (cont'd)

- What to do when the schedule slips
  - Despite allowances for slippage.....
  - The perils of delusion
  - Facing the music
  - Levelling with clients — mutual interests
  - Going into overdrive with existing staff
  - Rewarding the overdrive



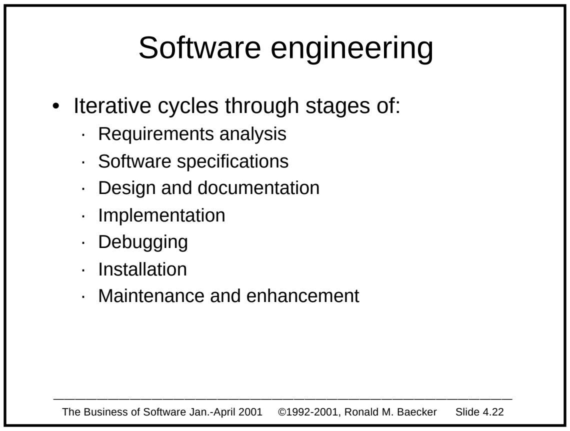
Software Prototyping and Software Engineering

- ***Use exploratory programming and software prototyping to experiment with new product ideas, including both functionality and interface; then employ more rigorous processes of software engineering to structure the carrying out of product development. (#25)***



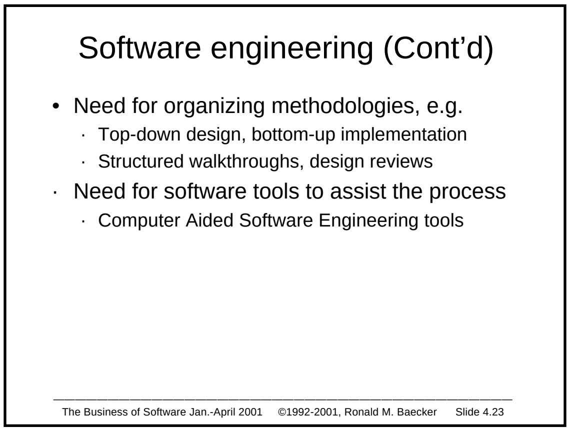
Exploratory programming and software prototyping

- Exploratory programming
- The need for iterative design
- The need for prototypes
- Roles for prototypes
  - Concept exploration
  - Communication with management
  - Communication with potential users



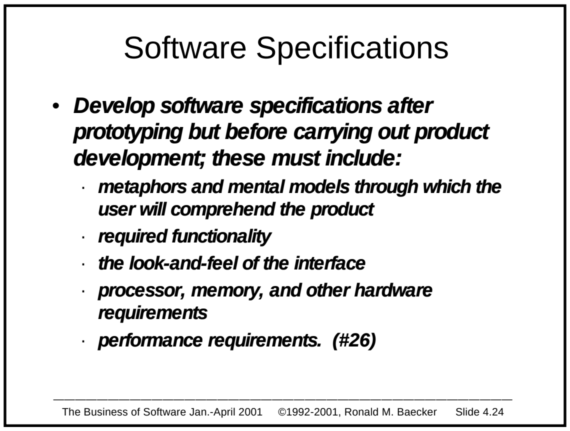
Software engineering

- Iterative cycles through stages of:
  - Requirements analysis
  - Software specifications
  - Design and documentation
  - Implementation
  - Debugging
  - Installation
  - Maintenance and enhancement



Software engineering (Cont'd)

- Need for organizing methodologies, e.g.
  - Top-down design, bottom-up implementation
  - Structured walkthroughs, design reviews
- Need for software tools to assist the process
  - Computer Aided Software Engineering tools



Software Specifications

- ***Develop software specifications after prototyping but before carrying out product development; these must include:***
  - ***metaphors and mental models through which the user will comprehend the product***
  - ***required functionality***
  - ***the look-and-feel of the interface***
  - ***processor, memory, and other hardware requirements***
  - ***performance requirements. (#26)***

Software User Interfaces

- ***A critical component of any piece of software is its user interface, which determines how it looks and feels to its users; effective interfaces are designed through a process that is user-centred, iterative, and multi-disciplinary. (#27)***

The interface

- What is the interface?
  - Where the human user and the computer meet
  - The look and feel of the software
- The importance of interfaces
  - Human time more important than machine time
  - Marketing appeal of good ergonomics and “user-friendly” systems
  - The success of the Macintosh
  - The success of the GUI
  - Beyond the GUI

User interface design

- User-centred design
  - User involvement, e.g., participatory design
  - Mental models / metaphors, e.g., desktop, checkbook, how system and interface understood by users
- Iterative design
  - Prototyping
  - User testing
- Multi-disciplinary design
  - Behavioural science
  - Graphic design

Software for all

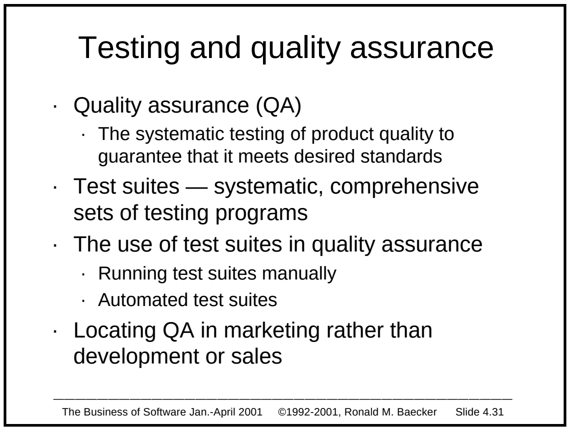
- Over 1 in 10 users have some disability
- Number increasing… demographics of aging
- Examples from other domains: curb cuts, closed captioning… relevance to everyone
- Try to increase potential user base in 6 ways
  - Motor access, e.g., through just the keyboard as well as keyboard and mouse
  - Sensory access, e.g., for colour-blind users
  - Different skill levels, e.g., novice, expert
  - Multi-cultural, linguistic, learning style, gender issues

Software Testing and Quality Assurance

- ***Software product quality must be systematically tested to guarantee that it meets desired standards; this includes alpha testing within one's own organization and beta testing at carefully selected customer sites. (#28)***

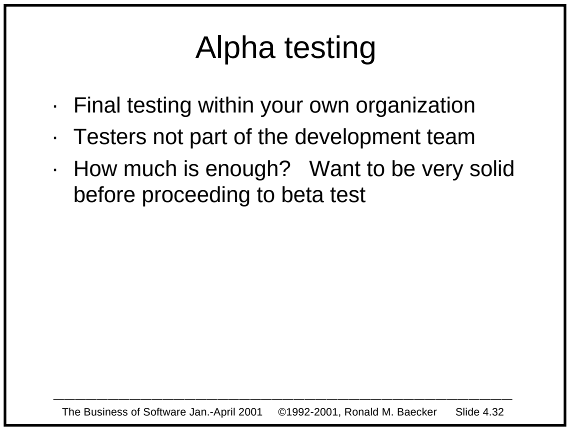
The need for testing

- Cost of fixing bugs grows dramatically, e.g.,
  - If 1 in design phase, perhaps 10 in implementation phase, perhaps 100 in the testing phase, perhaps 1,000-100,000 after shipment
- The “just one trivial fix” syndrome — “just a simple matter of programming”
- Serious bugs appear bizarre and illogical
- Example: Windows/NT — roughly 4,000,000 lines of code — roughly 100 testers
- Windows'95 over 10,000,000 lines of code



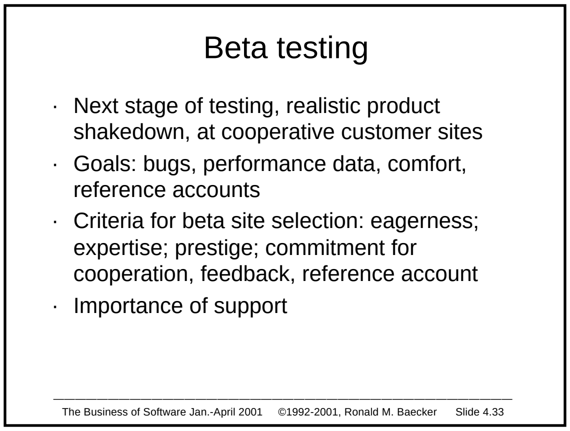
Testing and quality assurance

- Quality assurance (QA)
  - The systematic testing of product quality to guarantee that it meets desired standards
- Test suites — systematic, comprehensive sets of testing programs
- The use of test suites in quality assurance
  - Running test suites manually
  - Automated test suites
- Locating QA in marketing rather than development or sales



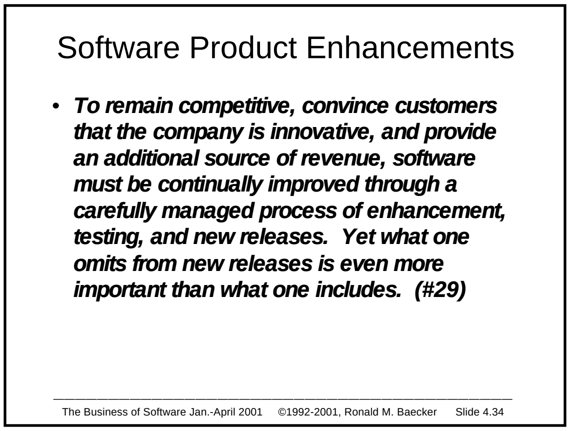
Alpha testing

- Final testing within your own organization
- Testers not part of the development team
- How much is enough? Want to be very solid before proceeding to beta test



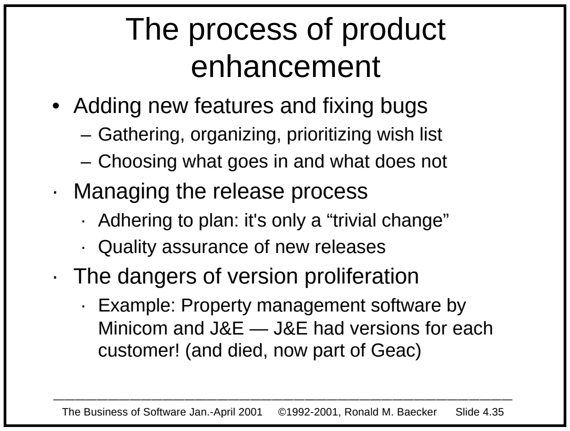
Beta testing

- Next stage of testing, realistic product shakedown, at cooperative customer sites
- Goals: bugs, performance data, comfort, reference accounts
- Criteria for beta site selection: eagerness; expertise; prestige; commitment for cooperation, feedback, reference account
- Importance of support



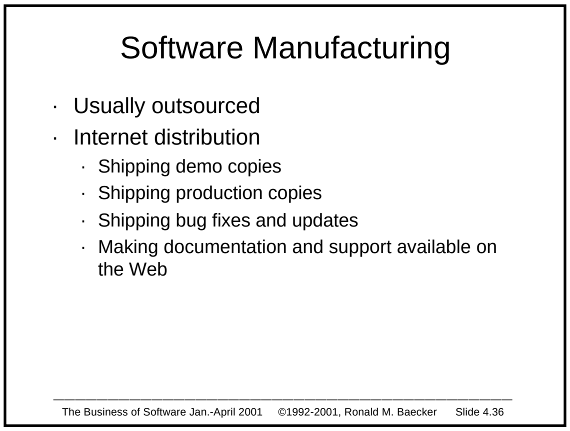
Software Product Enhancements

- ***To remain competitive, convince customers that the company is innovative, and provide an additional source of revenue, software must be continually improved through a carefully managed process of enhancement, testing, and new releases. Yet what one omits from new releases is even more important than what one includes. (#29)***



The process of product enhancement

- Adding new features and fixing bugs
  - – Gathering, organizing, prioritizing wish list
  - – Choosing what goes in and what does not
- Managing the release process
  - Adhering to plan: it's only a “trivial change”
  - Quality assurance of new releases
- The dangers of version proliferation
  - Example: Property management software by Minicom and J&E — J&E had versions for each customer! (and died, now part of Geac)



Software Manufacturing

- Usually outsourced
- Internet distribution
  - Shipping demo copies
  - Shipping production copies
  - Shipping bug fixes and updates
  - Making documentation and support available on the Web

Customer Support

- ***Competent and responsive customer support is essential in today's software marketplace. It provides additional revenue, and is required for effective sales to customers who are not computer specialists. It is a method of corporate and product differentiation, and a source of intelligence about your product and those of your competitors. (#30)***
- Example: WordPerfect support of 3000 — Corel reduced to <500 full-time support staff

The importance of support

- Complexity of software and systems
- End user customers not computer specialists
- Customers in distribution channel (e.g., dealers) overloaded with multiple inconsistent systems

The difficulty of support

- Differences in backgrounds and capabilities
- Lack of a common language
- Describing human-computer interactions at a distance
- Multiple interfaces, versions, & environments
- Psychological drain of being a full-time helper

Methods of support

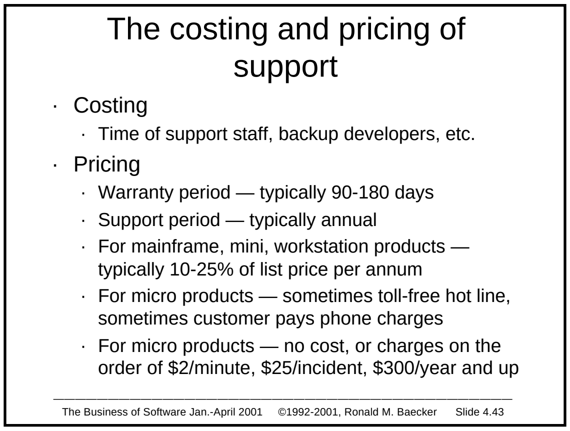
- Organizational
  - Dedicated support staff
  - Rotating staff, including management and technical staff (NO exceptions!)
  - Hierarchy of expertise
  - Ensuring followup — support logs, databases

Methods of support (cont'd)

- Technical
  - Telephone
  - Fax
  - Email
  - Web database of FAQs, updates, problem solutions, bug fixes (patches)
  - Direct machine connections
  - Just-in-time videos

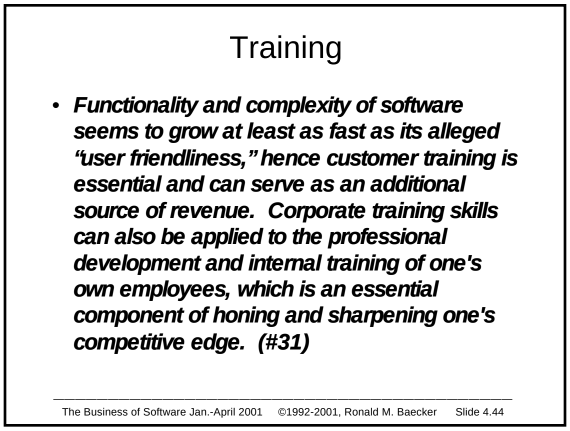
Learning from support

- Bugs in the program
- Features needed in the program
- Problems with the user interface
- Problems with the documentation
- What the customers really love
- What the competitors are offering
- What the competitors really have
- The key — Getting this information to marketing and development!



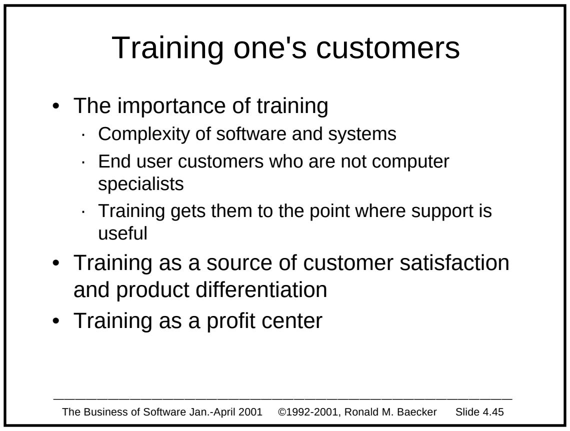
The costing and pricing of support

- Costing
  - Time of support staff, backup developers, etc.
- Pricing
  - Warranty period — typically 90-180 days
  - Support period — typically annual
  - For mainframe, mini, workstation products — typically 10-25% of list price per annum
  - For micro products — sometimes toll-free hot line, sometimes customer pays phone charges
  - For micro products — no cost, or charges on the order of $2/minute, $25/incident, $300/year and up



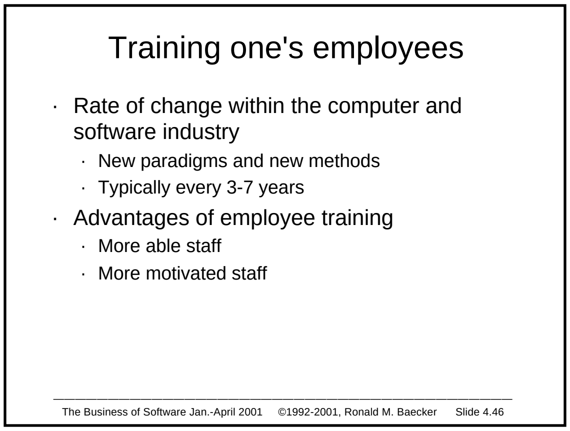
Training

- ***Functionality and complexity of software seems to grow at least as fast as its alleged "user friendliness," hence customer training is essential and can serve as an additional source of revenue. Corporate training skills can also be applied to the professional development and internal training of one's own employees, which is an essential component of honing and sharpening one's competitive edge. (#31)***



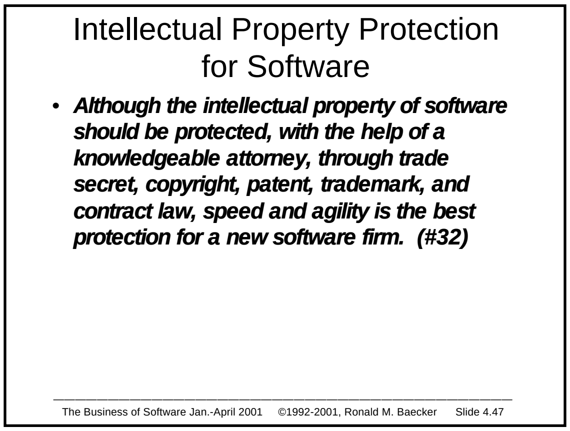
Training one's customers

- The importance of training
  - Complexity of software and systems
  - End user customers who are not computer specialists
  - Training gets them to the point where support is useful
- Training as a source of customer satisfaction and product differentiation
- Training as a profit center



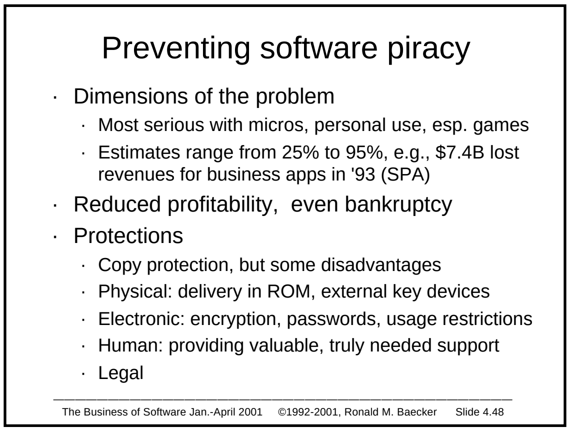
Training one's employees

- Rate of change within the computer and software industry
  - New paradigms and new methods
  - Typically every 3-7 years
- Advantages of employee training
  - More able staff
  - More motivated staff



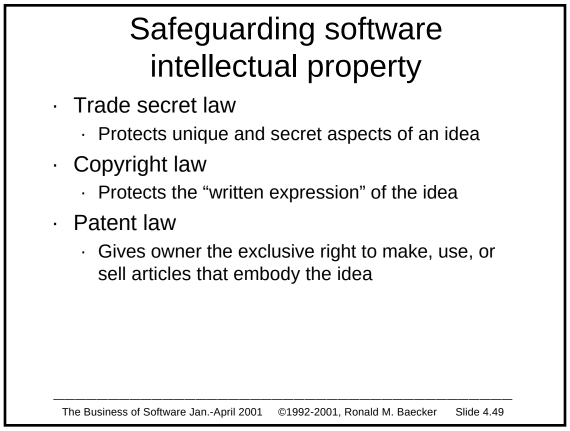
Intellectual Property Protection for Software

- ***Although the intellectual property of software should be protected, with the help of a knowledgeable attorney, through trade secret, copyright, patent, trademark, and contract law, speed and agility is the best protection for a new software firm. (#32)***



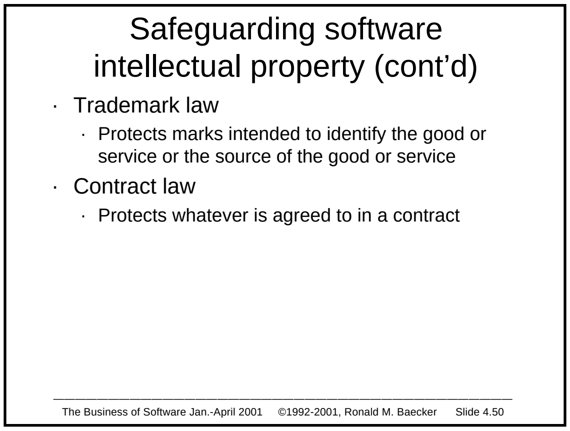
Preventing software piracy

- Dimensions of the problem
  - Most serious with micros, personal use, esp. games
  - Estimates range from 25% to 95%, e.g., $7.4B lost revenues for business apps in '93 (SPA)
- Reduced profitability, even bankruptcy
- Protections
  - Copy protection, but some disadvantages
  - Physical: delivery in ROM, external key devices
  - Electronic: encryption, passwords, usage restrictions
  - Human: providing valuable, truly needed support
  - Legal



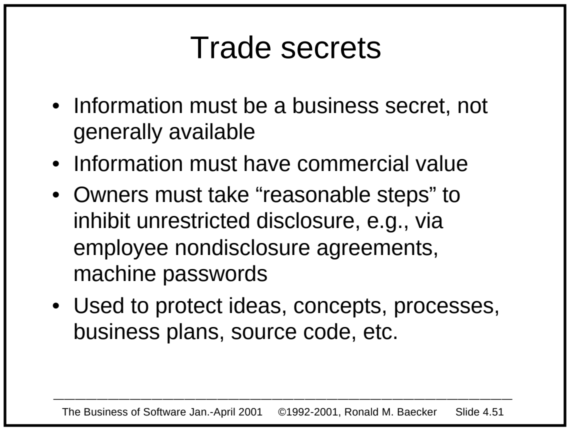
Safeguarding software intellectual property

- Trade secret law
  - Protects unique and secret aspects of an idea
- Copyright law
  - Protects the “written expression” of the idea
- Patent law
  - Gives owner the exclusive right to make, use, or sell articles that embody the idea



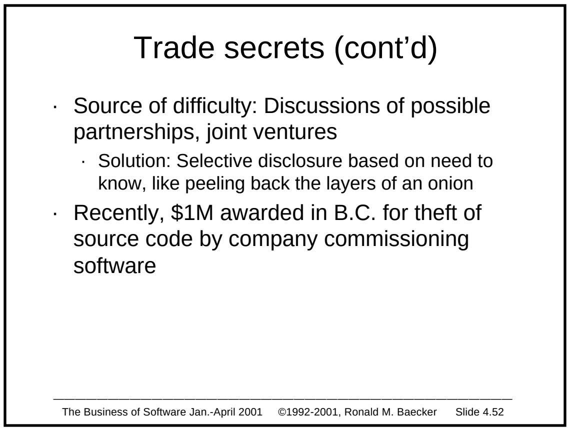
Safeguarding software intellectual property (cont’d)

- Trademark law
  - Protects marks intended to identify the good or service or the source of the good or service
- Contract law
  - Protects whatever is agreed to in a contract



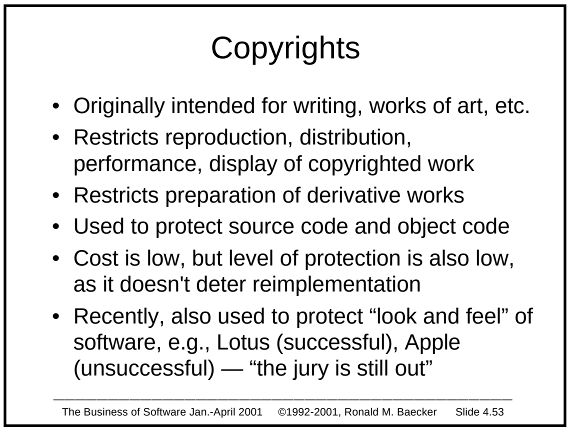
Trade secrets

- Information must be a business secret, not generally available
- Information must have commercial value
- Owners must take “reasonable steps” to inhibit unrestricted disclosure, e.g., via employee nondisclosure agreements, machine passwords
- Used to protect ideas, concepts, processes, business plans, source code, etc.



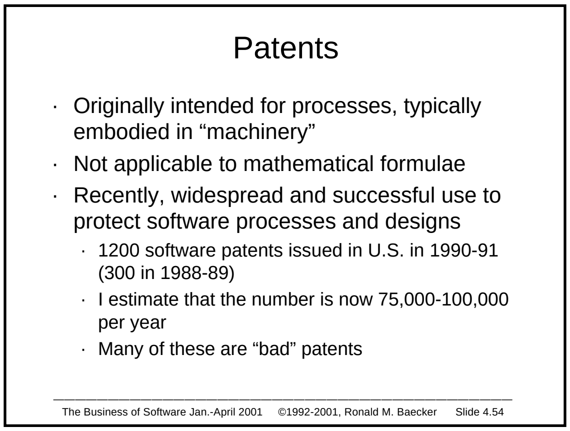
Trade secrets (cont’d)

- Source of difficulty: Discussions of possible partnerships, joint ventures
  - Solution: Selective disclosure based on need to know, like peeling back the layers of an onion
- Recently, $1M awarded in B.C. for theft of source code by company commissioning software



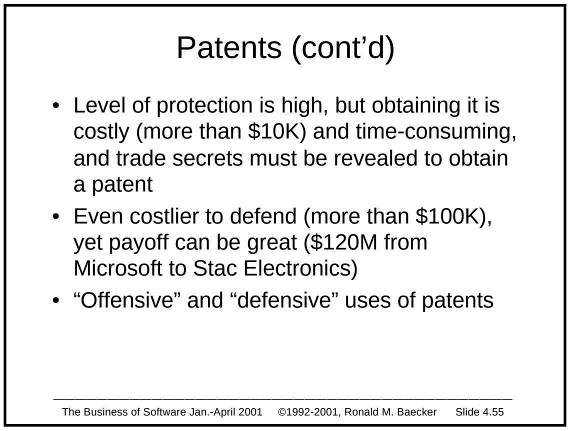
Copyrights

- Originally intended for writing, works of art, etc.
- Restricts reproduction, distribution, performance, display of copyrighted work
- Restricts preparation of derivative works
- Used to protect source code and object code
- Cost is low, but level of protection is also low, as it doesn't deter reimplementation
- Recently, also used to protect “look and feel” of software, e.g., Lotus (successful), Apple (unsuccessful) — “the jury is still out”



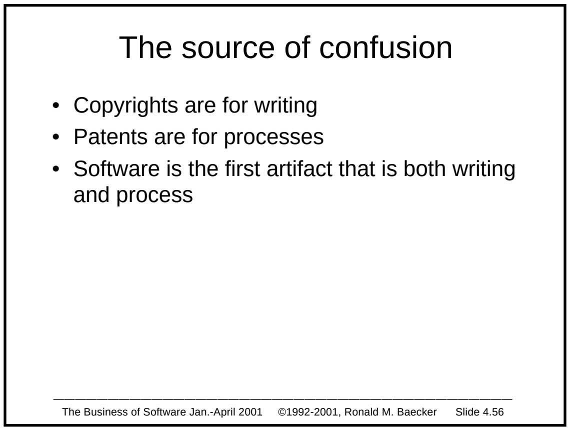
Patents

- Originally intended for processes, typically embodied in “machinery”
- Not applicable to mathematical formulae
- Recently, widespread and successful use to protect software processes and designs
  - 1200 software patents issued in U.S. in 1990-91 (300 in 1988-89)
  - I estimate that the number is now 75,000-100,000 per year
  - Many of these are “bad” patents



Patents (cont’d)

- Level of protection is high, but obtaining it is costly (more than $10K) and time-consuming, and trade secrets must be revealed to obtain a patent
- Even costlier to defend (more than $100K), yet payoff can be great ($120M from Microsoft to Stac Electronics)
- “Offensive” and “defensive” uses of patents



The source of confusion

- Copyrights are for writing
- Patents are for processes
- Software is the first artifact that is both writing and process