



Starting a Product

Session 3

Course ICT Entrepreneurship

Prof.dr. Sjaak Brinkkemper

Dr. Slinger Jansen

What is Product Software?



A software product is defined as

*a **packaged configuration** of software components
or a software-based **service**,
with auxiliary materials,
which is **released** for
and **traded** in a specific market.*

So, what is

1. The application
2. The market
3. The software components
4. The package

Software start-up study



- Erran Carmel:
 1. A Process Model for Packaged Software Development
 2. Time-to-completion in software package startups

- Study of 12 starting product companies software

- Investigation of **successful approaches** in the acceleration of software production

- In tailor-made software world: **"Software is always late"**

Papers on methods for Product Software companies



This presentation:

- Carmel, E. and Sawyer, S., (1998) "Packaged Software Teams: What Makes Them So Special?," *Information Technology & People*, 11(1), 6-17
- Carmel, E. (1995). Time-to-completion factors in packaged software development. *Information & Software Technology* Vol. 37, Num. 9, 1995, pp. 515-520

Other:

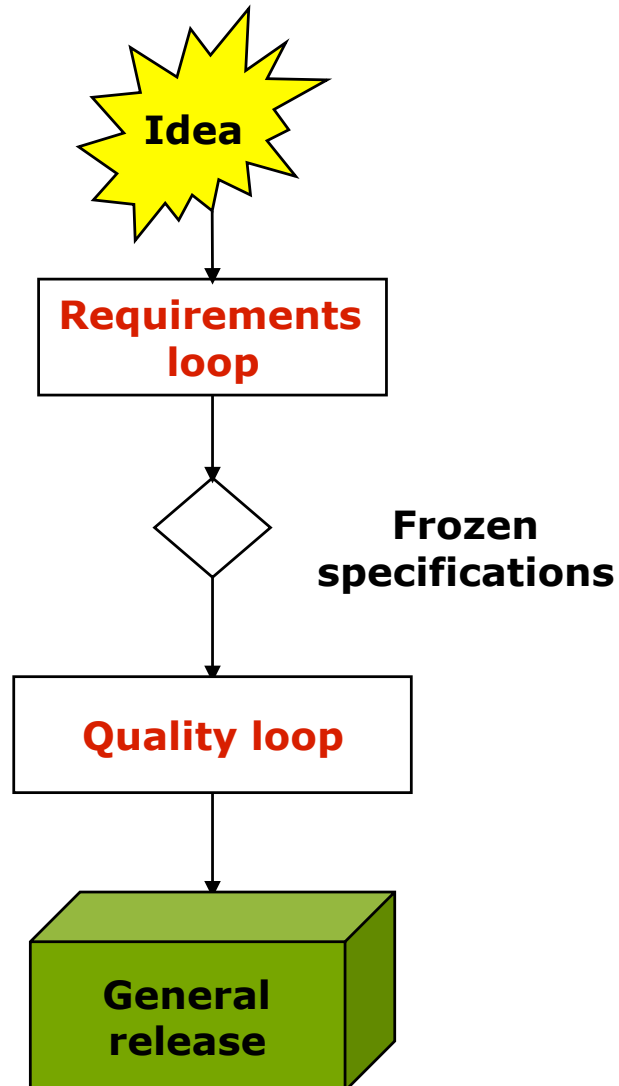
- Carmel, E. (1995). Cycle Time in Package Software Firms. *J. Prod Innov. Manag.*, 1995:12:110-123
- Carmel, E. (1996). American hegemony in packaged software trade and the "culture of software". *The Information Society*, Volume 12 number 4
- Sawyer, S. and Guinan, P., (1998) "Software Development: Processes and Performance," *IBM Systems Journal*, 37(4), 552-569
- Erran Carmel and Shirley Becker. A Process Model for Packaged Software Development. *IEEE Transactions on Engineering Management*, vol.42, no. 1, pp.50-61, 1995.
- Helms, R.W., Brinkkemper, S., Oosterum, J. van, & Nijs, F. de (2005). Knowledge Entry Maps: structuring of method knowledge in the IT industry. In Y Kiyoki, H Kangasslo, H Jaakkola, & J Henno (Eds.), *Proceedings of the 15 European Japanese Conference on Information Modelling and Knowledge Bases* (pp. 12-25). Amsterdam: IOP Press.
- Klooster, Marnix, Sjaak Brinkkemper, Frank Harmsen, Gerard Wijers, Intranet Facilitated Knowledge Management: A Theory and Tool for Defining Situational Methods. In: A. Olivé and J.A. Pastor, *Advanced Information Systems Engineering. Proceedings of the 9th International Conference CAiSE'97*, Barcelona, Spain, Lecture Notes in Computer Science 1250, pp. 303-317, Springer Verlag, June 1997.

How nature protects weak products



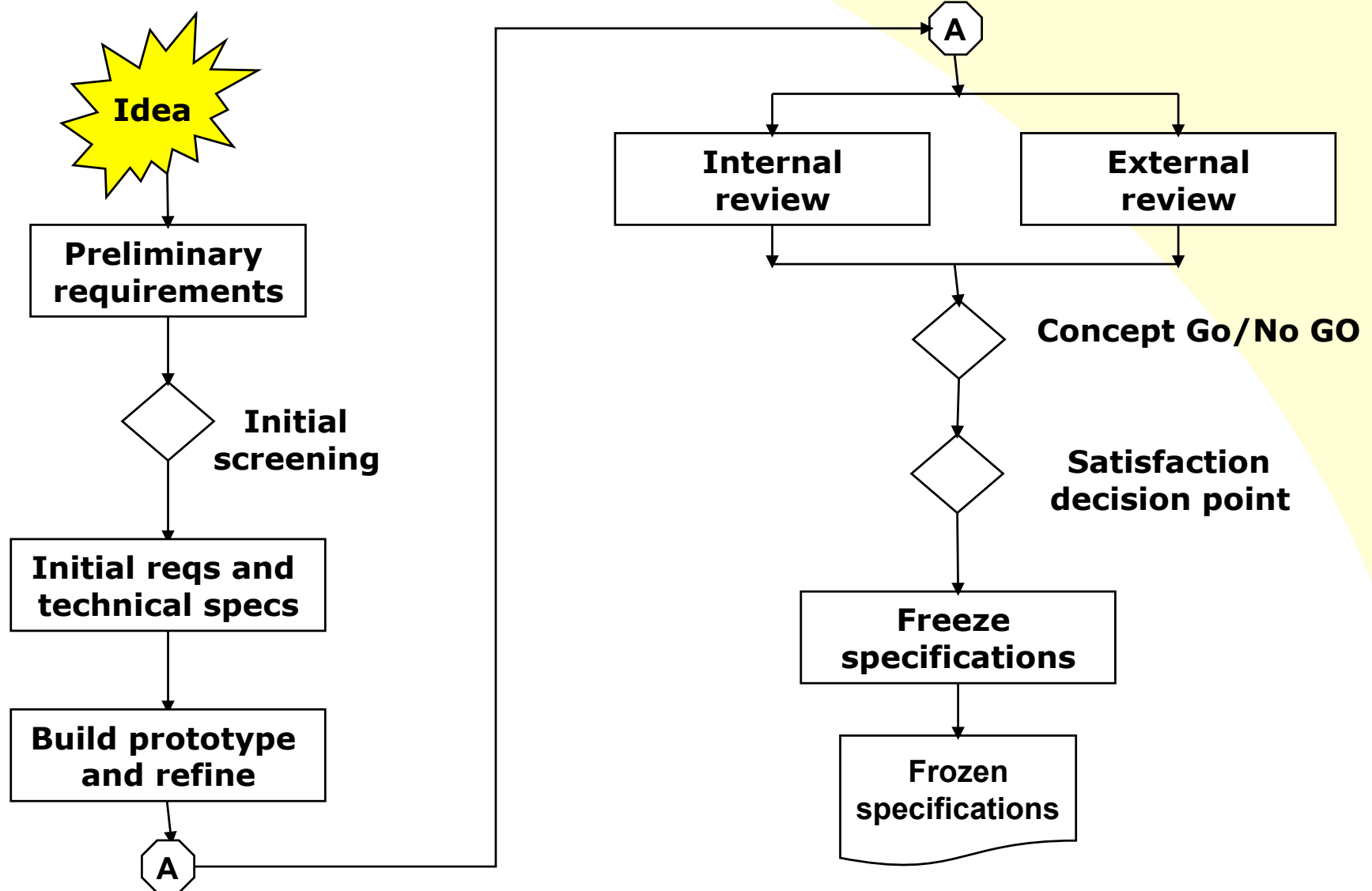
**"How Nature
Protects Weak
Products"**

Process model for product software

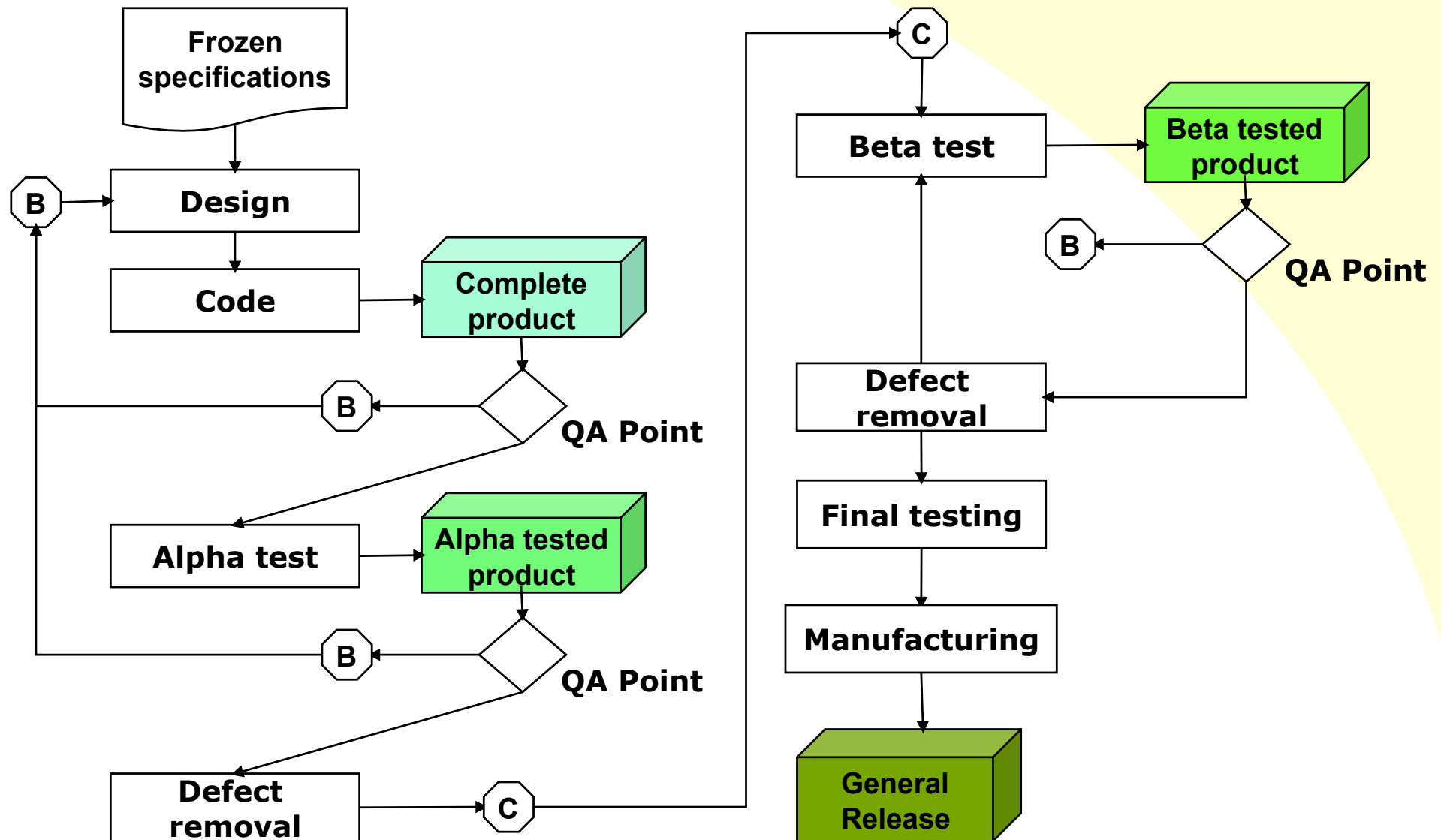


- Formulating idea and prototyping
- Design and code until quality threshold is reached

Requirements loop



Quality loop



Background



- **Time-based competition** in business
- **Time-to-completion** (time-to-market) is:
measure of calendar time from beginning of product development to product release

Release of new car model: 36 months

Release of new printer: 22 months

Release of new software product: ?

- **Justification:**
 - Fast pay-back of R&D investment
 - Faster cycles of product renewal
- What about product software?

Productivity in software



Definitions

Productivity = effort per time unit * person

Effort \cong Lines of Code (LOC)

Standard productivity: 20 LOC /personday

In order to be predictable for future releases it is imperative to collect data from the beginning

- **Professional** atmosphere
- Not used **against** persons
- **Precision** of data is not an issue

Productivity accelerators



Literature review

1. Development method (life cycle model):

- water-fall: complete stages
- spiral: start from core release
- incremental: build architectural chunks
- evolutionary: adapt according changing wishes

2. Development team

- individual characteristics
- group characteristics

3. Resources

- labor
- tools, infrastructure,
- office space, environment, social

Productivity accelerators (2)



4. Project Management:

- managing people, milestones, resources and tasks
- style and tools

5. Risk Analysis

- identifying, assessing, and managing risks

6. Incremental Innovation

- changes from one product to the next

1 and 2 in software research

Others from general product development

Research method



- 12 product software firms
- Revenue: 1 – 12 M\$
- US: Virginia – Maryland area
- Products for PC
- Selected because of:
 - Breadth of product offering
 - Product complexity
 - Age of firm
 - Minimal success: products were purchased
- Interviews with key developers (18)
- Survey to team members (30 out of 48)

Firm characteristics



| Firm # | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|--------------|---------------------|---|---|---|---|---|---|---|---|---|----|----|----|
| Application | Business/Office | | | √ | | √ | | | | | √ | | |
| | Specialized | √ | √ | | | | | | | √ | | | |
| | Systems / Utilities | | | | √ | | √ | | | | | √ | √ |
| | Graphics | | | | | | | √ | √ | | | | |
| Distribution | Mass market | | | √ | √ | | √ | | | | | √ | |
| | Wide | √ | | | | √ | | | √ | | √ | | √ |
| | Narrow/high end | | √ | | | | | √ | | √ | | | |

Development time (in months)



| Firm | First product | Subsequent product(s) |
|-------------|---------------|-----------------------|
| 1 | ? | 24 |
| 3 | 26 | In development |
| 4 | 5 | 24 |
| 5 | 13 | None |
| 6 | 36 | 11,12,13,13 |
| 7 | ? | 6,8,12,2 |
| 8 | ? | 19,10,9 |
| 9 | 48 | 18 |
| 11 | 24 | 18,36 |
| 12 | 38 | 9 |
| Avg. | 26.0 | 14.3 |

Data were **rough estimates**
Multiples of 12

Hardly a time strategy, when asked

Time-to-completion unknown

Deadlines not so important

Competitive pressures

Market pressures

Acc. 1: Development method



| Development method | Degree of implementation | # Firms |
|---------------------------|---------------------------------|----------------|
| Specifications | None | 2 |
| | Informal, brief, a few pages | 7 |
| | Formal specifications | 3 |
| Design | Informal, brief, minimal | 8 |
| | Formal and distinct stage | 4 |
| Prototyping | No data | 1 |
| | None | 3 |
| | Little or rarely | 1 |
| | Prototyping actively used | 7 |
| Testing | Informal or non-standard | 6 |
| | Formalized phases | 6 |

Development method findings



- **Accidental life-cycle model:**
 - custom system for one client is turned into software product
 - Methodical approach after release 1.0
- **Entrepreneurial nature of firm**
 - Shortage of money
 - Critical events
 - Method not in culture
- **Importance of software architecture**
 - Not on paper
 - Known by every team member
- **Use of Prototyping**
 - Proof-of-concepts, Mock-up, demos
 - Decreases time-to-completion
 - Increases quality
- **Overlapping phases**
 - Poor planning (or perhaps more agile?)

Growth stage model

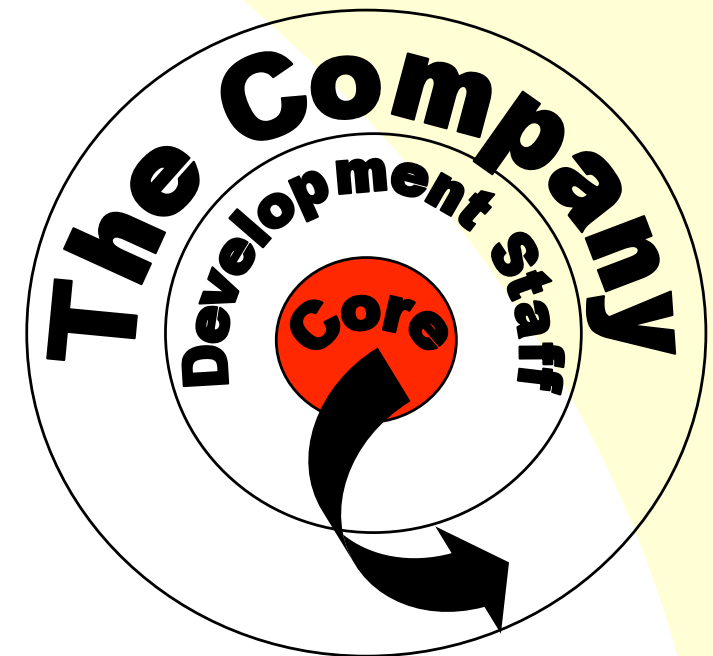


| | Idea | Version 1.0 | 10 employees | 20 employees |
|------------------------------|-----------------|---|---------------------|------------------------------------|
| | Inception | First steps | Transition | Rapid growth |
| Development method | Ad hoc | Necessary steps Config. Mgmt Error sheets | | Elements of formal method |
| Organization | Core team forms | Non-development staff added | | Additional development staff added |
| Quality assurance | Minimal | Some | | Formalization begins |
| # firms in this study | 0 | 5 | 3 | 4 |

Acc 2: Development Team



- Essential factor
- Core team
 - Homogeneous
 - Highly motivated
 - Highly experienced
 - History of working together
 - Core team
 - Architect
 - Principal programmers
 - Cohesive
 - Dense communication
 - Whole duration



Size of team related to age



| Firm | Years since release of first product | Total number of employees | | |
|-------------|--------------------------------------|---------------------------|-------------------|-----------|
| | | Company | Development Staff | Core team |
| 5 | <1 | 6 | 3 | 3 |
| 12 | 2 | 8 | 3 | 3 |
| 8 | 2 | 8 | 4 | 3 |
| 3 | 2 | 5 | 3 | 3 |
| 9 | 3 | 13 | 7 | 4 |
| 2 | 3 | 20 | 6 | 6 |
| 1 | 4 | 10 | 6 | 3 |
| 4 | 5 | 45 | 5 | 4 |
| 6 | 6 | 70 | 45 | 4 |
| 11 | 6 | 57 | 35 | 6 |
| 7 | 7 | 50 | 11 | 4 |
| 10 | 9 | 20 | 11 | 5 |
| Avg. | 4.2 | 26 | 11.6 | 4 |

Core team findings



■ Size

- About 4
- No growth during growth of company

■ Composition

- Most between 20 and 30
- Male, American
- Varying education: from high school to PhD
- Very high motivation levels
- Stock options
- Good wages

Quotes



- There are **six of us** who know the intimate details of what the product does
- There are at most 100 pages of design ... and no one knows where that is anymore – it's in peoples' brains
- The core team members were in **one room** – like one brain – no phone calls, no interruptions
- All we need is at most 4-5 developers, even if we get to be a \$10 million company. We couldn't survive with any more.

More on the core team



- **Experience**
 - Computer science
 - Application domain essential
 - 13 years experience in programming
- **Duration of co-work**
 - Before joining core team
- **Team structure**
 - Amorphous
 - Some with strong player
 - Others no dominant leader



Acc. 3: Resource allocation



- **Labor is a flexible resource**
 - Hire temp help
 - Relocate project members
- **Finding: no labor added**
 - Extra time of team
 - 87% work more than 56 h/week
 - 47% work more than 71 h/week



Capital investments



- **Little investments in tools**
 - Non used design tools
 - Less than 1000 USD / year * developer
 - Freeware/shareware utilities
 - Trial versions from customers/vendors
- **In-house developed utilities**
 - Test software
 - More investment than on outside tools
- **Distrust in others' software**

Acc. 4: Project Management



■ Usage of PM

- 6 companies use formal PM tools, scheduling, timelines.
- 1 (of these) used PM tool all of the time
- Others use nothing

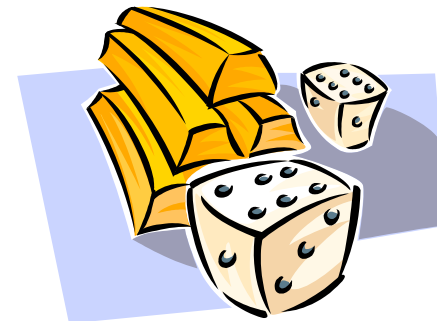
■ Meetings

- 1 or 2 meetings per week for key developers
- Coordination in less formal channels

Acc. 5: Risk Analysis



- No **systematic** approach to risk analysis
- Risk management at the **end** of project
 - Reduction of time to completion
 - Quality versus functionality
- No formal policies



Acc. 6: Incremental Innovation



- Many firms applied **incremental** innovation
- **Product families**
 - Closely related products
 - Derivation from other products
 - New versions regular time intervals
 - Manageable updates (service packs)
- **Niche** innovation in their markets
 - Established new niches

Additional accelerator: Quality



- Testing is complex and consumes time and resources
- Testing was **compromised**
- **Quality assurance**: making sure that a product fulfills certain prescribed quality properties
 - QA was absent
 - Weak methodology
 - Lost in discussions on time and functionality

Lessons from this study



- **Core team is the accelerator** with the most impact on the time-to-completion
 - Recognized as an asset in companies
 - Cross functional, small size, selective staffing, motivation, full-time involvement
 - History of working together
 - Homogeneous, highly motivated, in-depth experience
- **Other accelerators present**
 - Use of extra effort at the end of project
 - Building similar products
 - Remaining accelerators are weak
- **Weak awareness of time-to-completion**
 - Deadlines were soft
 - Hardly a strategy for development speed
 - Poor recall of time spend on project

Implications for software start-ups



- **Founders of start-ups**
 - Creation of successful core team
 - One talented developer not enough
 - Interesting product ideas and features not enough
 - Well-mixed, experienced team

- **How about Europe and Asia?**
 - Cultural aspects
 - Working attitude
 - Seniority versus egalitarian