Factor 10x - Super Efficient Startups

Business patterns for servicing many customers with a small team

- This talk is about the usual stuff:
  - War stories from a start-up
  - Software reuse
  - Disparate roles and their communication in developer teams, like most agile methodologies

- But there are special twists:
  - Extreme efficiency (because there were only two guys with awkward CVs)
  - We use a heavy mix of business + tech language to explain how this works
 - Nice pictures & awkward jokes

- Limitations - These ideas only work for:
  - B2B or business software
  - Relatively slow changing market
  - Market niche with high-sophistication software

- Down-Side:
  - Living by your limits - e.g. an "ugly" UI
  - Not moving particularly fast (not having the "fast mover" advantage but having the "survivor" advantage)
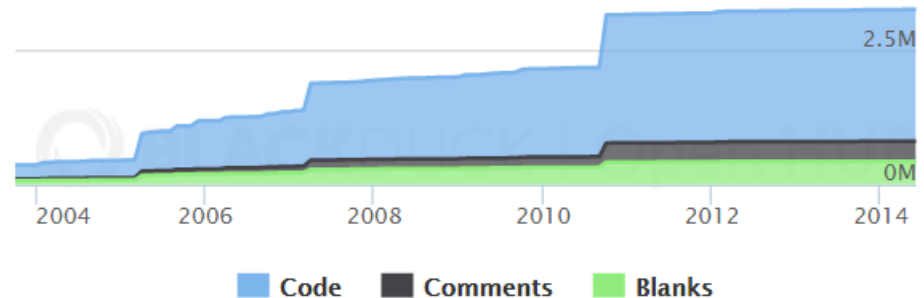
Quickly about]project-open[

Facts about]**project-open**[

- Multi-Project Management Web Application
- ~6,000 Customers in 30 countries
- Some really big customers
- ~200,000 Users connecting to a]po[server every week

- ~250 functional "packages"
- ~1,000 Web "pages"
- 1.7 M LoC (2.5M LoC – 30% JavaScript)
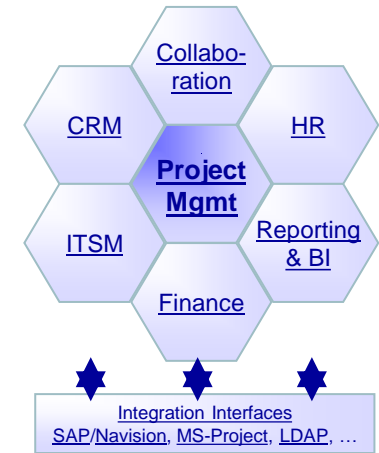
- **Developed and run mainly by two guys**

- Philip Greenspun:
  "There is no [Web] problem that 5 developers couldn't solve in 3 months"

- CraigList:
  Few guys

- WhatsApp:
  200 million users with 50 Employees in 2013

What is]**project-open**[? – Old-Style Page

Welcome System Administrator | 1 user online | Context Help | Report a bug on this page | **Log Out**

Search [ ] Go

Home ▾ | Users ▾ | Projects ▾ | Companies ▾ | Timesheet ▾ | Absences ▾ | Reporting | XoWiki | Admin ▾ | My Settings ▾

Summary | Files | Finance | Tasks | Expenses | Gantt | Resources | Gantt Editor | MS- Warning | Audit Trail s | Risks | Timesheet Interval | Earned Value | EVA | Cost | Sharepoint | Rule Audit

Side Menu

Project Base Data

Project name	Special Machine Half (2014-05-15)
Parent Project	
Project Nr	2014_0010
Client	ABC Consulting
Project Manager	
Project Type	Gantt Project
Project Status	Open
Start Date	2014-09-20
Delivery Date	2015-11-27 17:00
On Track Status	◌
Program	Special Machines Program
Presales Probability	30
Presales Value	146400
Project Priority	5 - Average

Edit

Project Baselines

	Baseline	Baseline Type	Baseline Status	Baseline Creation Date
	No Baselines Available			

Delete Baselines ▾ Apply

• Create a new Baseline

Forum Items

P	Type	Subject	Due	Own	Ass	Status	

There are no active items.

Mark as Read ▾ Apply

Timesheet Tasks

Showing tasks 1 to 10 Next Page

	Task Name	Start	End	Pln	Bll	Lg	%	UoM	Members
☐	⊟Special Machine Half (2014-05-15)	14-09-20	15-11-27	451.1	451.1			Hour	CC, DD, DA, PP, TT, SD, JD
☐	⊞Milestones	15-05-27	15-11-27	0.0	0.0			Hour	TT
☐	⊟Phase Konstruktion	15-05-31	15-07-02	72.7	72.7			Hour	CC, PP, SD
☐	⊞Schnittstelle Kunde	15-05-31	15-07-02	72.7	72.7			Hour	CC, PP, SD
☐	⊞Phase Fertigung	15-06-14	15-08-27	132.0	132.0			Hour	DD, DA, TT
☐	⊟Phase Montage	15-08-09	15-10-10	170.0	170.0			Hour	DD, TT, JD
☐	Aufbau Mechanik	15-08-09	15-09-26	120.0	120.0			Hour	JD:43%
☐	Aufbau Elektrik	15-08-28	15-09-24	30.0	30.0			Hour	DD:19%
☐	⊟Integration Fremdfertiger	15-09-27	15-10-10	20.0	20.0			Hour	TT

Project Gantt Schedule

	Year	2014		2015			
	Quarter	Q3	Q4	Q1	Q2	Q3	Q4

⊟ Special Machine Half (2014-05-15)
⊞ Milestones
⊞ Phase Konstruktion
⊞ Phase Fertigung
⊞ Phase Montage
⊞ Phase Inbetriebnahme
⊞ Installation Kunde

Project Gantt Resources

	Year	2015			
	Quarter	Q2	Q3	Q4	Σ
⊞ Consultant Consultant	Σ	Σ	4%		1%
⊞ Developer Developer	Σ	Σ	16%	2%	4%
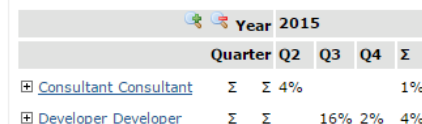
What is]**project-open**[? - HTML5 Page

What is]project-open[?

]open[
PROJECT

Diagram

Hexagon diagram:
- Collabo-ration
- CRM
- HR
- **Project Mgmt** (center)
- ITSM
- Reporting & BI
- Finance

Integration Interfaces
SAP/Navision, MS-Project, LDAP, …
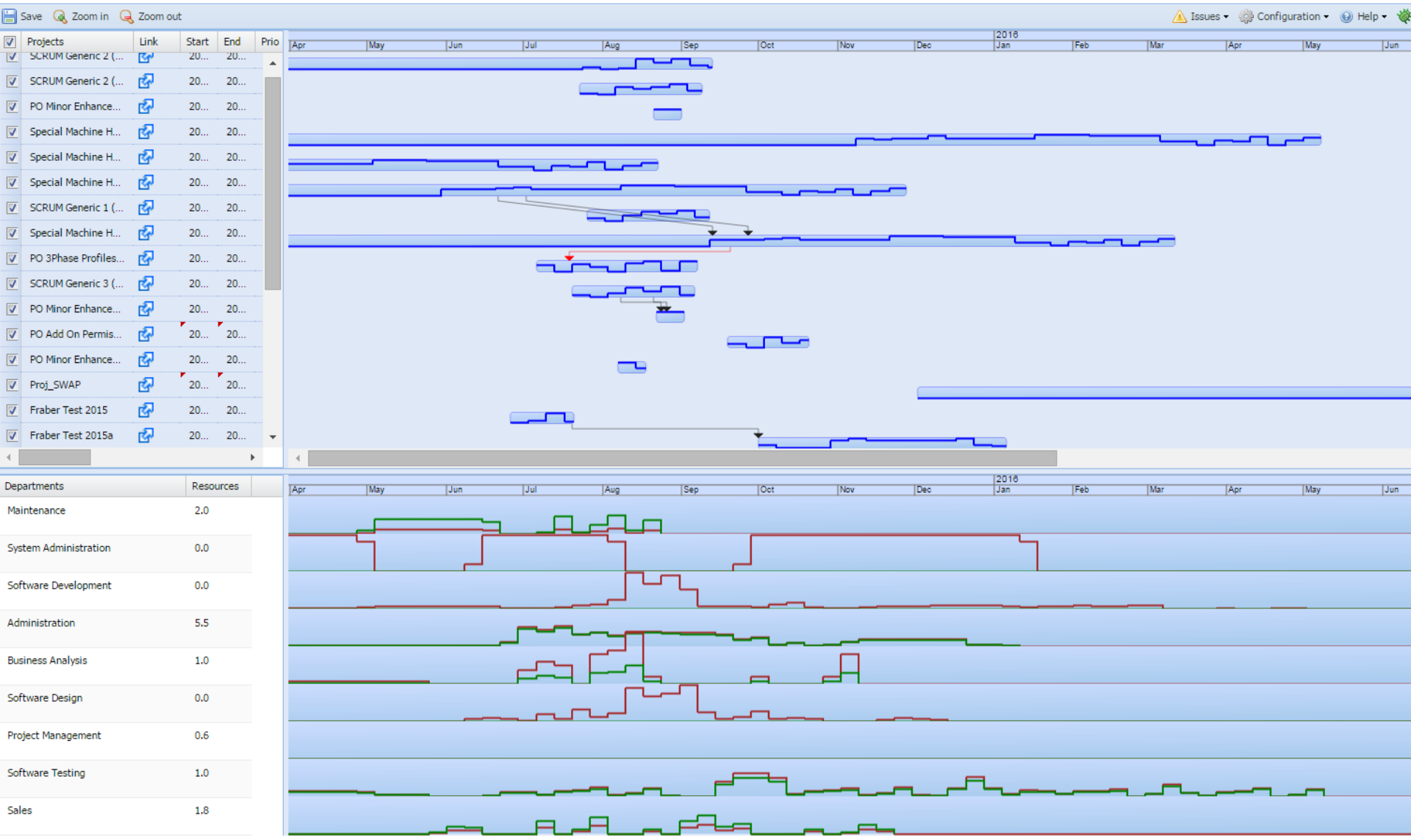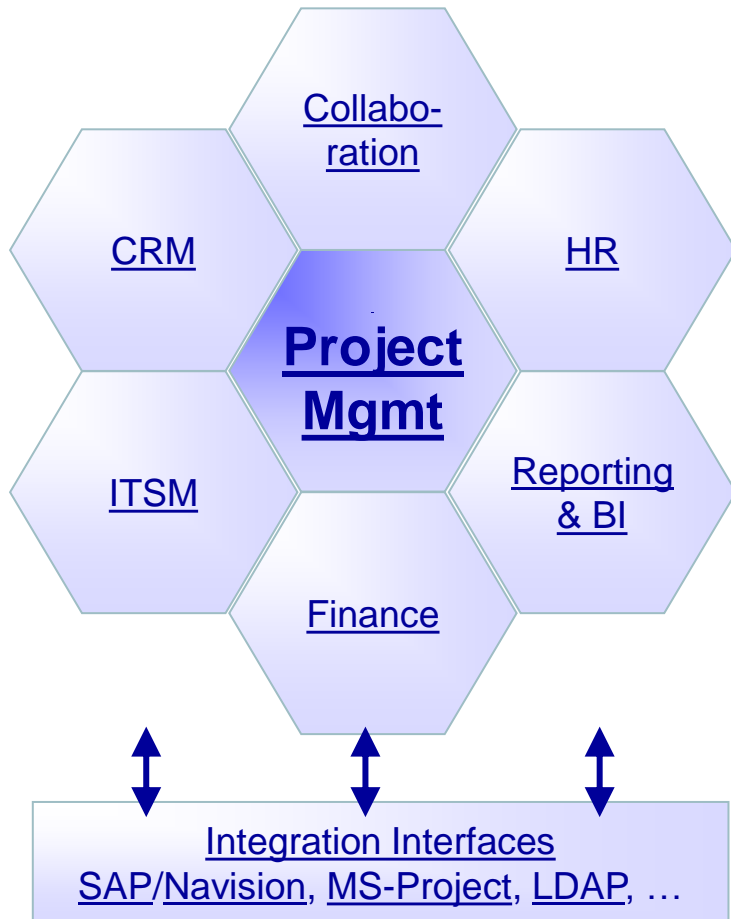
What is]project-open[?

-]po[is an open source project management application with a focus on finance and collaboration.
- It's main purpose is to coordinate the work of project managers, project members and stake holders across multiple projects.
- Target organizations are companies or corporate departments that need to track finances, tasks, and resources for a large number of concurrently running projects.
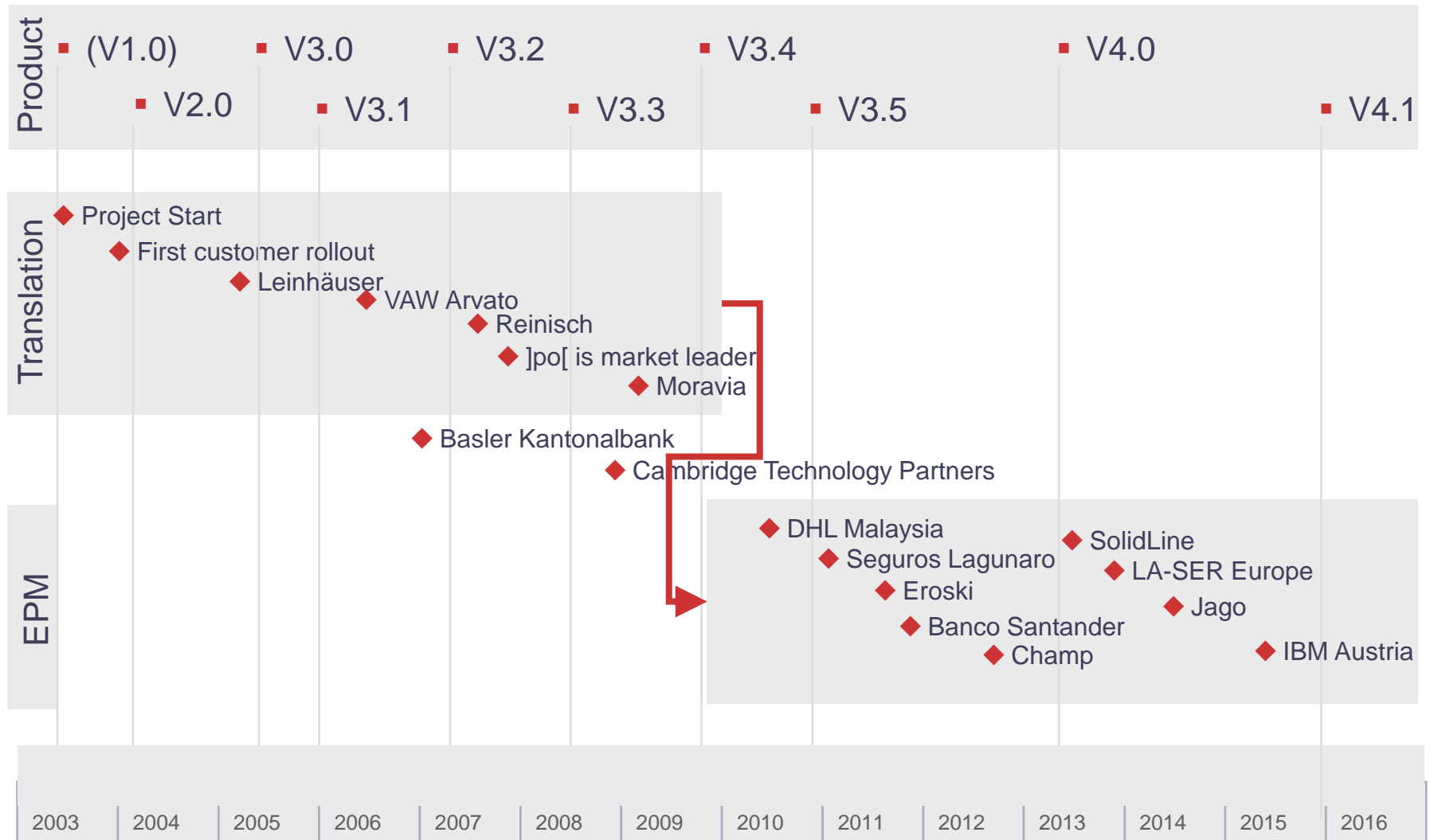
Customers

- ~6.000 companies worldwide use]po[in production.
- Top #3 European Bank (~1.500 users)
- Basler Kantonalbank (~300 users)
- Eroski Supermarkets (~300 users)
- Cambridge Technology Partners (~250 users)
- Seguros Lagunar Aro (~150 users)
- DHL Malaysia (~150 users)
- VAW arvato S.L. (Bertelsmann) (~60)

Achievements

- Optaros: "3 stars – mature" enterprise readiness rating
- Heise.de: "Germany's #3 open-source ERP/CRM application"
- Ohloh: "#1 open-source web application with 3.000.000 lines of code"
- SourceForge: Ranking between #50-200 and 80.000 downloads of]po[V3.5

]po[Time Line

Product

- ■ (V1.0)
- ■ V2.0
- ■ V3.0
- ■ V3.1
- ■ V3.2
- ■ V3.3
- ■ V3.4
- ■ V3.5
- ■ V4.0
- ■ V4.1

Translation

- ◆ Project Start
- ◆ First customer rollout
- ◆ Leinhäuser
- ◆ VAW Arvato
- ◆ Reinisch
- ◆]po[is market leader
- ◆ Moravia
- ◆ Basler Kantonalbank
- ◆ Cambridge Technology Partners

EPM

- ◆ DHL Malaysia
- ◆ Seguros Lagunaro
- ◆ Eroski
- ◆ Banco Santander
- ◆ Champ
- ◆ SolidLine
- ◆ LA-SER Europe
- ◆ Jago
- ◆ IBM Austria

| 2003 | 2004 | 2005 | 2006 | 2007 | 2008 | 2009 | 2010 | 2011 | 2012 | 2013 | 2014 | 2015 | 2016 |

Factor 10x

- Code Reuse & Efficiency
  - Open-Source Model
  - "Catapult Start"
  - "Raw Diamonds"
  - "MBA Codes"
  - About Libraries
  - Functional Style
  - "Sunflower Model"

- Business Strategy
  - "Bridging the Chasm"
  - "Project vs. Product"
  - "Product Management & Strategy"
  - "Zero Sales"
  - "Zero Support"

- Coding
  – Open-Source Development Model
  – Open-Source Reuse Model
  – Open-Source Security Model
- Business
  – Open-Source Distribution Model
  – Open-Source Sales Model
  – Open-Source Support Model
  – Open-Source Product Management Model

]project-open[and OpenACS
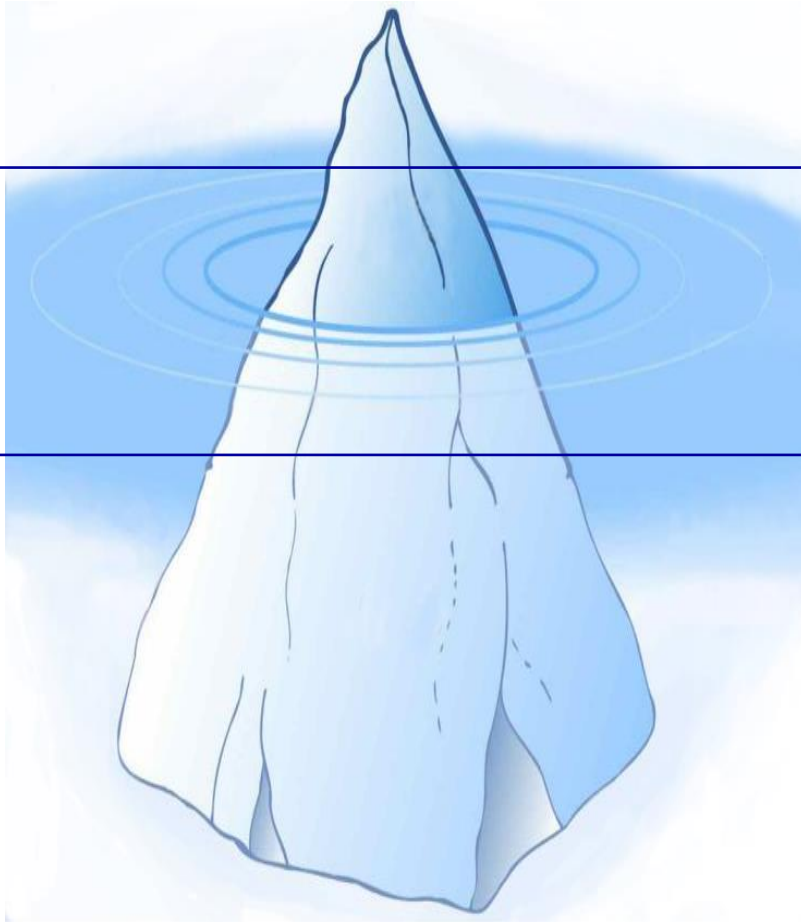
]project-open[
- 2+1 Owners, Σ60 years in IT
- Product driven

Developers/Partners
- ~10 Freelancers, ~15 Partners
- Sales driven

OpenACS/AOLserver
- 10.000 Members, ~100 active developers
- Driven by ~10 small web companies
- OpenACS Products: dotLrn e-learning platform
- Technology driven

Business

- No idea what the customers really want
- No idea what the customers would pay for
- No idea about the competition
- No idea what features you'll need to implement in the future

Technology

- No experience yet with the chosen framework
- No experience yet if the infrastructure really scales
- No experience with debugging the production system

- **Get your Minimum Viable Product to the market ASAP**
  - Start with an existing (open-source) application/framework
  - Start with an alpha-customer who urgently needs the functionality and provides you functional clues

- Negative Formulation
  - Reduce your lack of market knowledge by using a real-world customer
  - Reduce your lack of app application by using a proven application/framework
  - Reduce the lack of trust from your alpha and other customer with fast results

- The]project-open[Case
  - We chose the OpenACS 3.4 framework, the most mature Web framework in 2003
  - OpenACS was proven to work with a million concurrent users
  - OpenACS already had an embryonic "intranet" package with projects and hours
  - The alpha-customer was the company of my then-wife. It had an urgent need to better manage project resources

- Results
  - Three month after the start, the first PMs managed resources with]project-open[
  - Thanks to the new application, the translation company made €20.000 in additional benefits
  - We had our first success story
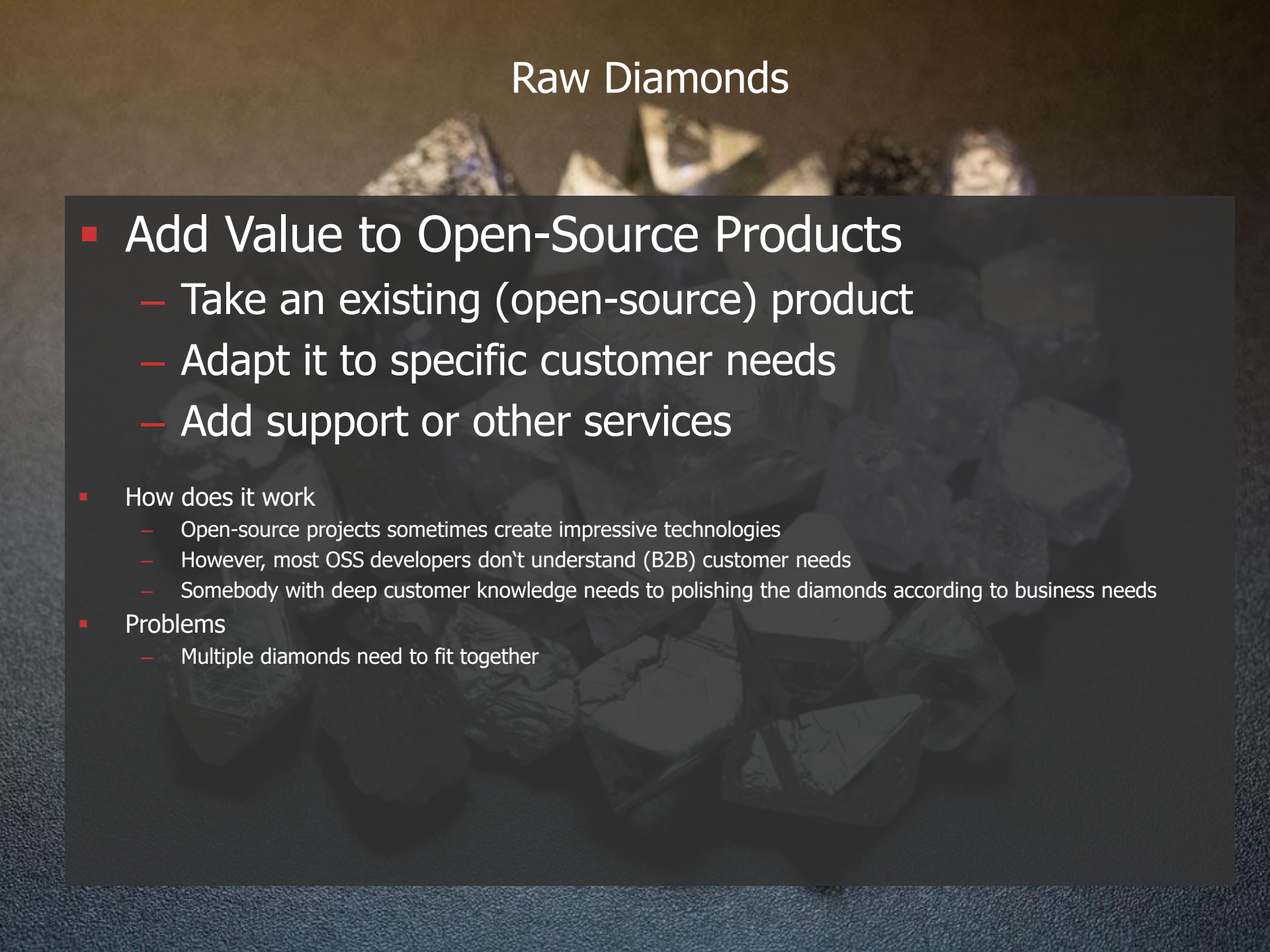 - This story convinced a lot of similar customers…

Examples

- Wordpress or similar (Web Application)
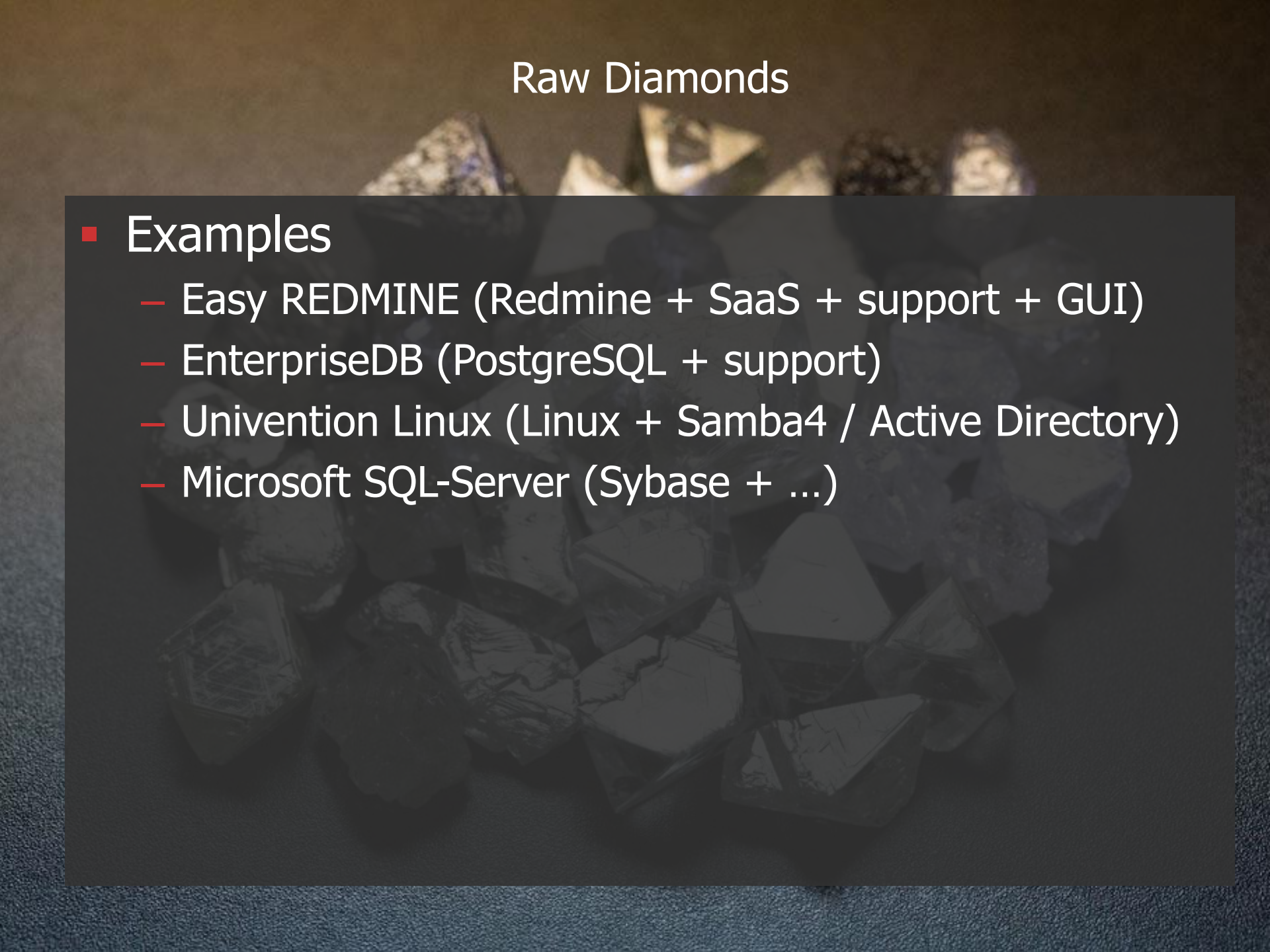- Java Technology stack + some OSS application
- Sencha ExtJS/Touch for mobile app

- **Add Value to Open-Source Products**
  - Take an existing (open-source) product
  - Adapt it to specific customer needs
  - Add support or other services

- How does it work
  - Open-source projects sometimes create impressive technologies
  - However, most OSS developers don't understand (B2B) customer needs
  - Somebody with deep customer knowledge needs to polishing the diamonds according to business needs
- Problems
  - Multiple diamonds need to fit together

- Examples
  - Easy REDMINE (Redmine + SaaS + support + GUI)
  - EnterpriseDB (PostgreSQL + support)
  - Univention Linux (Linux + Samba4 / Active Directory)
  - Microsoft SQL-Server (Sybase + …)

Libraries/Frameworks

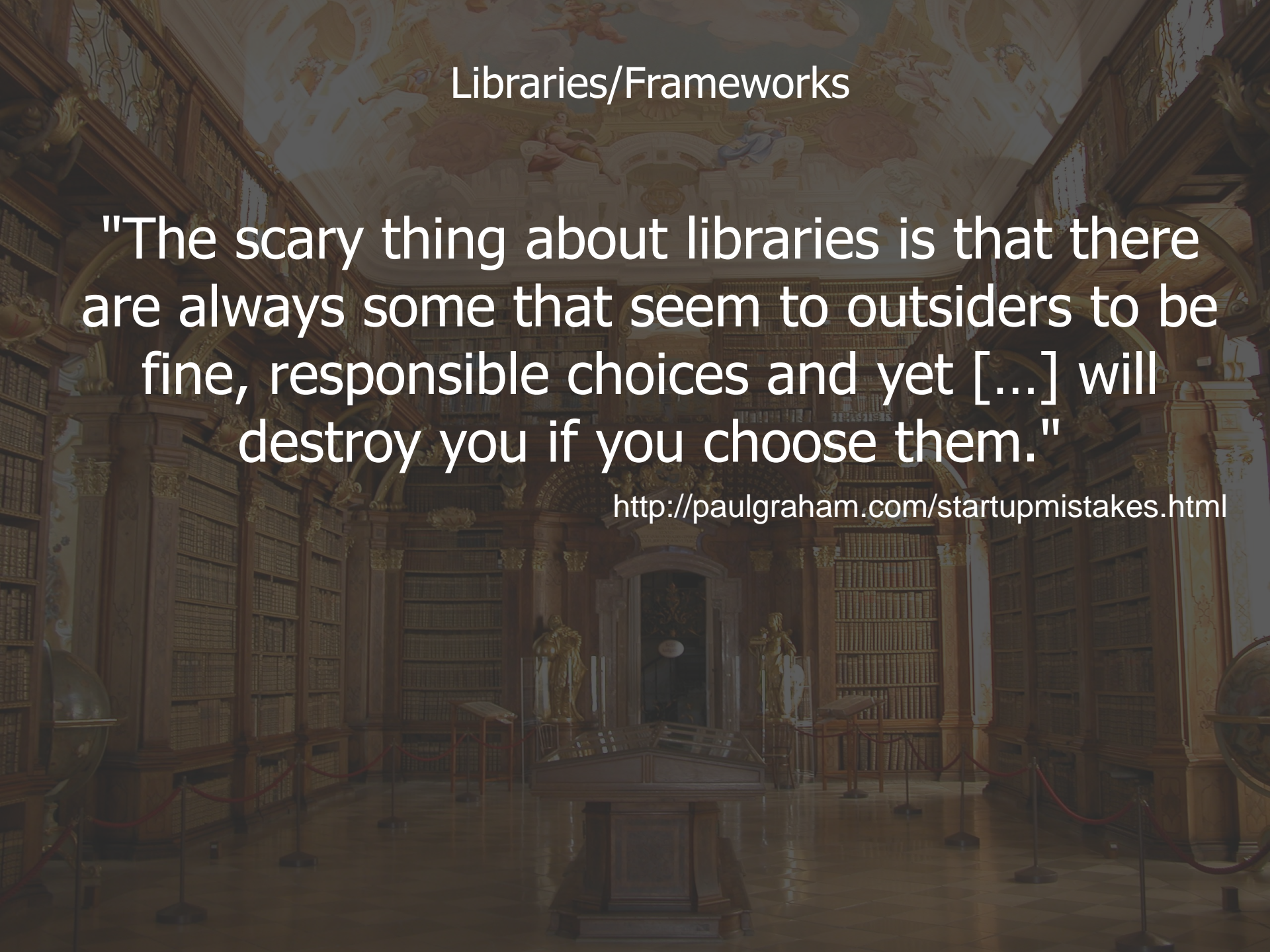"I found this cool library.
It's brand new and easy to use"

Death verdict for the startups

"The scary thing about libraries is that there are always some that seem to outsiders to be fine, responsible choices and yet [...] will destroy you if you choose them."
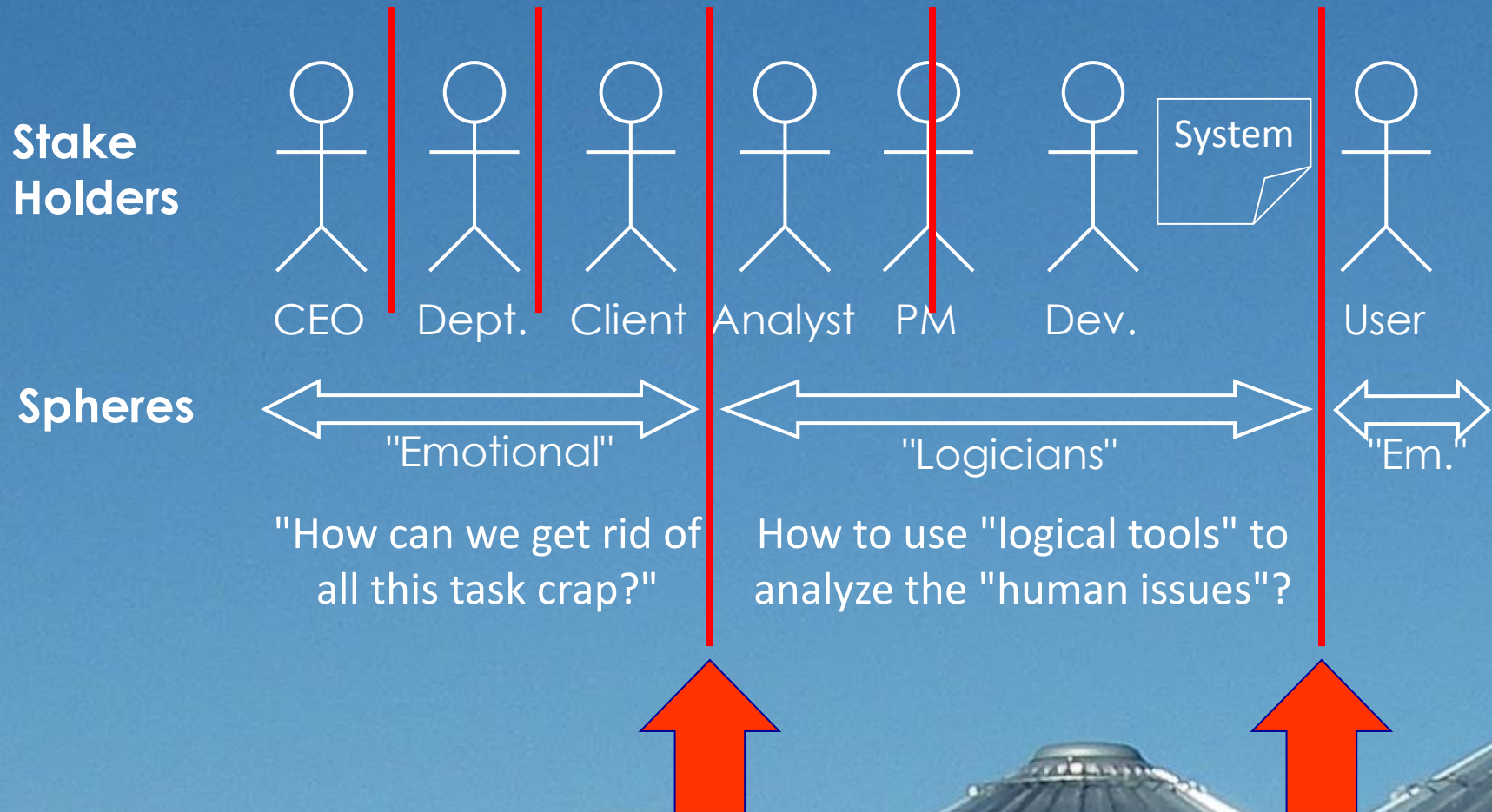
http://paulgraham.com/startupmistakes.html

Libraries/Frameworks Checklist

Libraries cover "commodity" functionality.
It should cover **ALL** commodity functionality:

- Localization
  - UTF-8
  - Number, date & currency formatting
  - Localization workflow
- Debugging Tools
- Peformance
  - Clustering, scaling, parallelization
  - Performance debugging
- "Horizontal Functionality"
  - Permissions
  - Web APIs (REST, JSON, XML, …)
  - Workflow
  - Email notifications & parsing
  - Search-Engine
- Team collaboration
- Scripting
- … (just have a look at some better ones)

- GUI Widgets
- GUI Forms
- Diagrams
- …

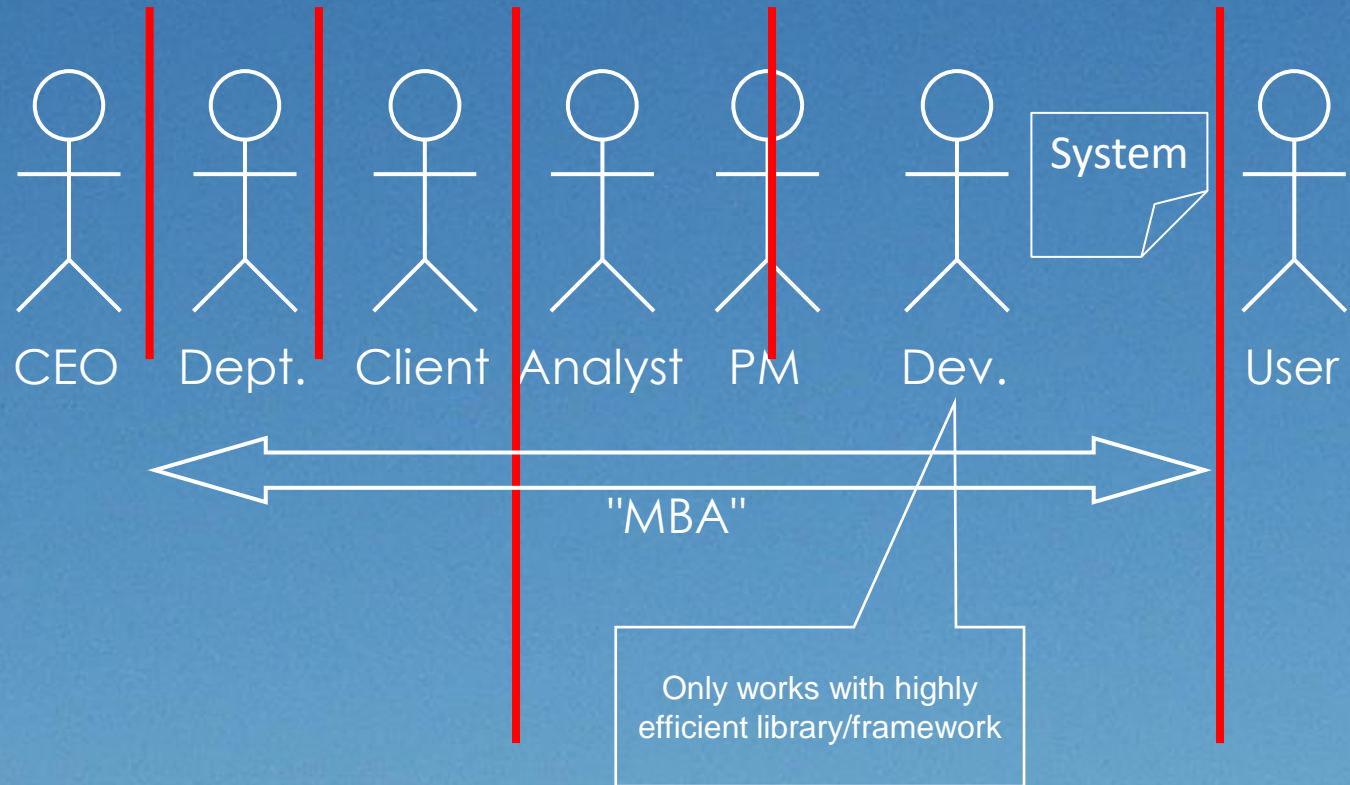- Actually working system in a similar area like yours

"MBA Codes" -]po[Product Development Pipeline
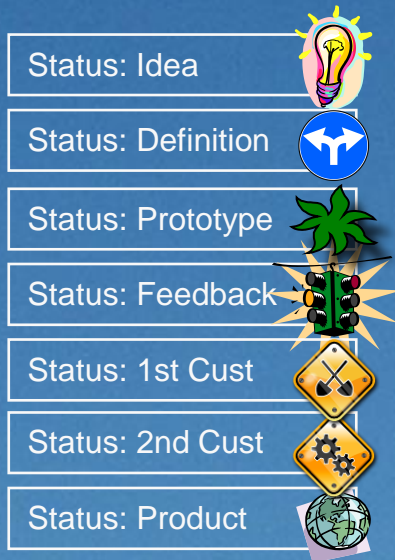
Status: Idea

Status: Definition

Status: Prototype

Status: Feedback

Status: 1st Cust

Status: 2nd Cust

Status: Product

- Results from brainstorming, conference, ...
- Product definition as PowerPoint slides
- "Dynamic Mockup"
- Show mockup to customers, conferences, ...
- Implementation driven by customer's input
- Re-Implementation with product in mind
- Cleanup and productification
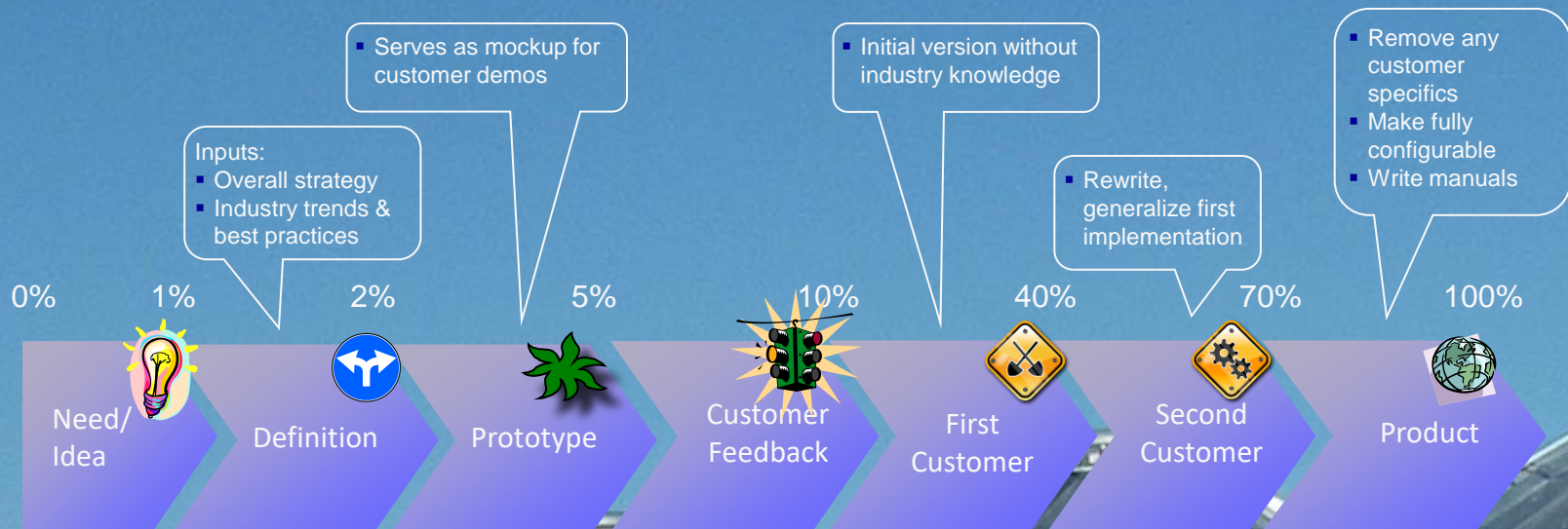
Inputs:
- Overall strategy
- Industry trends & best practices

Serves as mockup for customer demos

Initial version without industry knowledge

Rewrite, generalize first implementation

Remove any customer specifics
- Make fully configurable
- Write manuals

0%	1%	2%	5%	10%	40%	70%	100%
Need/Idea	Definition	Prototype	Customer Feedback	First Customer	Second Customer		Product

λ– What if I told you functions are values?

Window 1: `~/www`

File Edit Options Buffers Tools Tcl Help

```tcl
set table ""
db_foreach group "select * from cc_users" {
    append table "<tr><td>$email</td><td>$first_names</td><td>$last_name</td></tr>\n"
}
```

`-UUU:**--F1   simple-table.tcl    All L1      (Tcl) ------------`

Window 2: `~/www`

File Edit Options Buffers Tools Help

```
<master>
<table>@table;noquote@</table>
```

`-UUU:----F1   simple-table.adp    All L2      (Fundamental) ----`

Web page navigation:
Home | Bugs | Forum | Users | Programs | Projects | Companies | Workflow | Timesheet | Absences | Notes
Tickets | Finance | Milestones | Ideas | Conf Items | Resource Management | Reporting | Dashboard | Indicators
Expenses | Admin | My Settings

Side Menu

sysadmin@tigerpond.com	System	Administrator
va.hels@project-open.com	Draco	Draculevich
bbreimann@tigerpond.com	Barbara	Breimann
apicard@wanadoo.fr	Angelique	Picard
ddeveloper@tigerpond.com	David	Developer
daniel_df@gmx.de	Daniel	Dampfast

THE FUNCTIONAL WAY IS THE RIGHT WAY
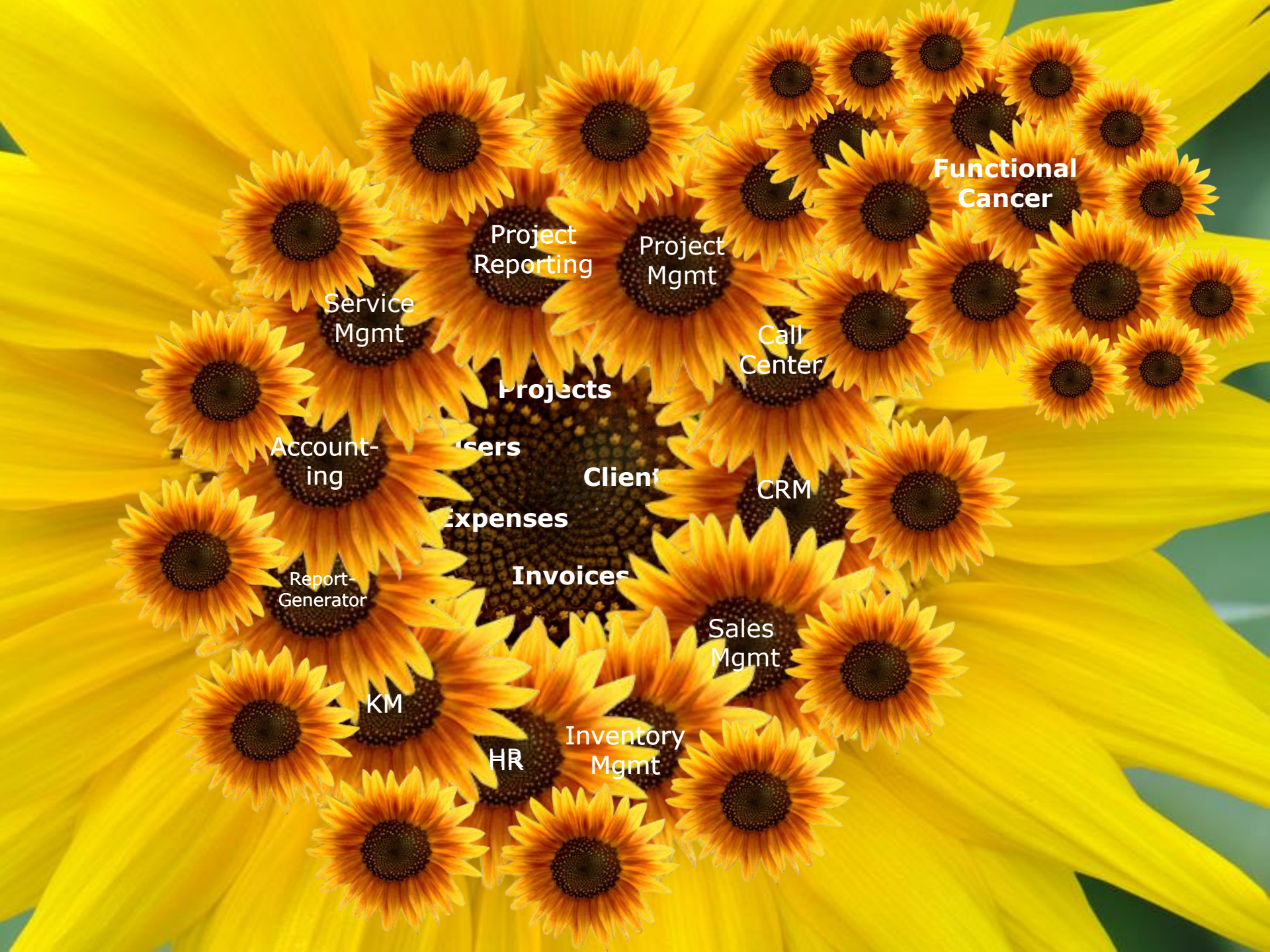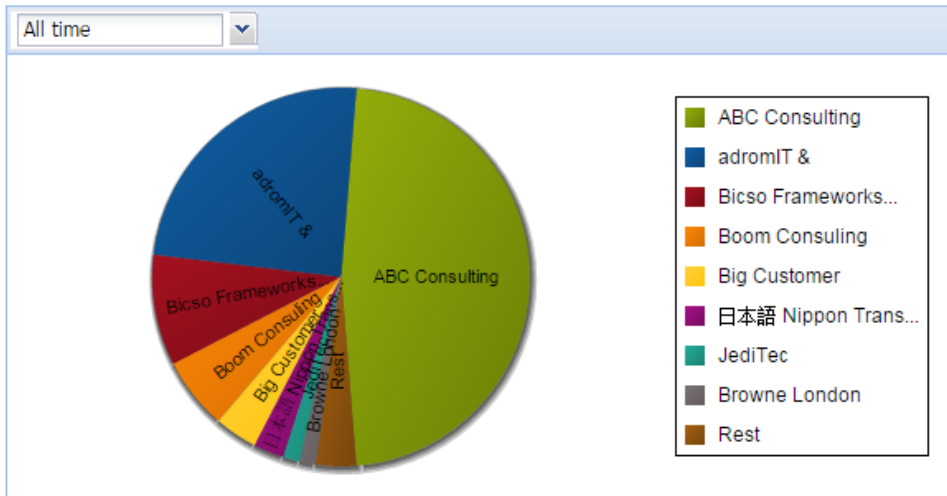
- Generic
- "Core"
- Slow changing
- High-Quality
- Configurable
- Low Biz Value
- "Product"

- Specific
- Non-Core
- Fast changing
- Low-Quality
- Customizable
- High Biz Value
- "Project"

Two Examples

All-time customer revenues

DynField Maintenance Screen
("dynamic fields")

Two Examples
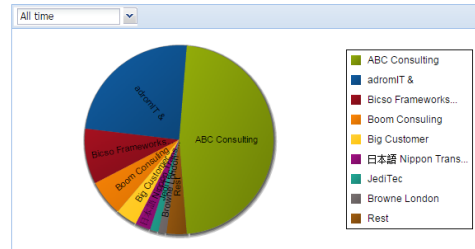
Criteria/Dimension	Customer Revenue Widger	DynField Maintenance
Technology	Sencha HTML5	"ad_form"
Prettyness	High	Very low
Genericity	High	Very high
Distance from "Core"	High	Low (this is part of the Core...)
Interconnection	Low	High
Change Speed	Medium	Very low
Customer Value	High	High
Strategic Value (for Cust)	Low	Low
"Curedness" (like ham)	Low	High
Algorithmic Complexity	Medium	Low
Usage Frequency	High	Very low
VIP Level	High	Very low
Alignment with Product Strategy	High	High

Open-Source vs. Closed Source
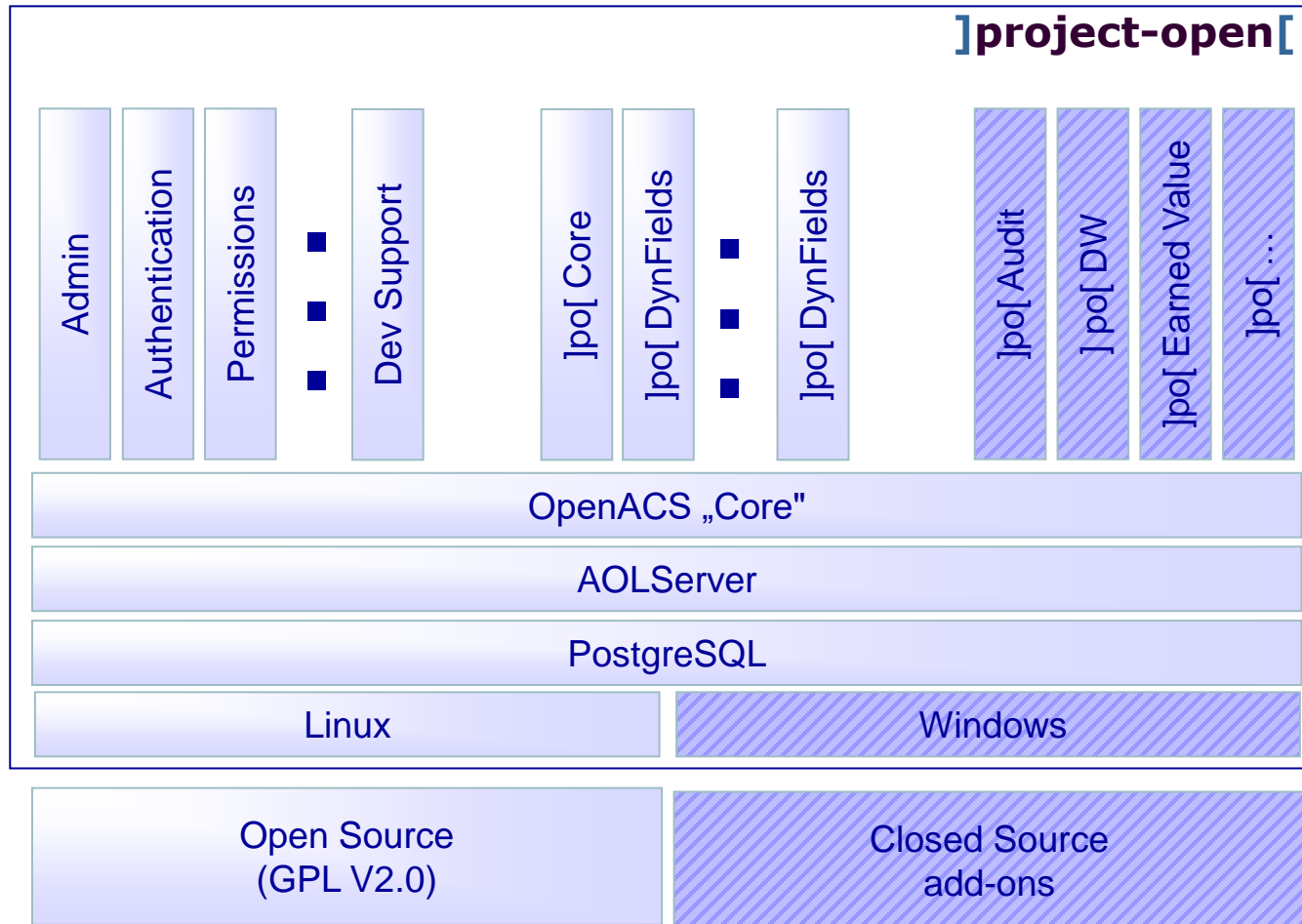
- Code Reuse & Efficiency
  - Open-Source Model
  - "Catapult Start"
  - "Raw Diamonds"
  - "MBA Codes"
  - About Libraries
  - Functional Style
  - "Sunflower Model"

- Business Strategy
  - "Bridging the Chasm"
  - "Project vs. Product"
  - "Product Management & Strategy"
  - "Zero Sales"
  - "Zero Support"

Bridging the Chasm

Geoffrey A. Moore

Bridging the Chasm -]po[Time Line

]po[PROJECT open[

Product

- (V1.0)
- V2.0
- V3.0
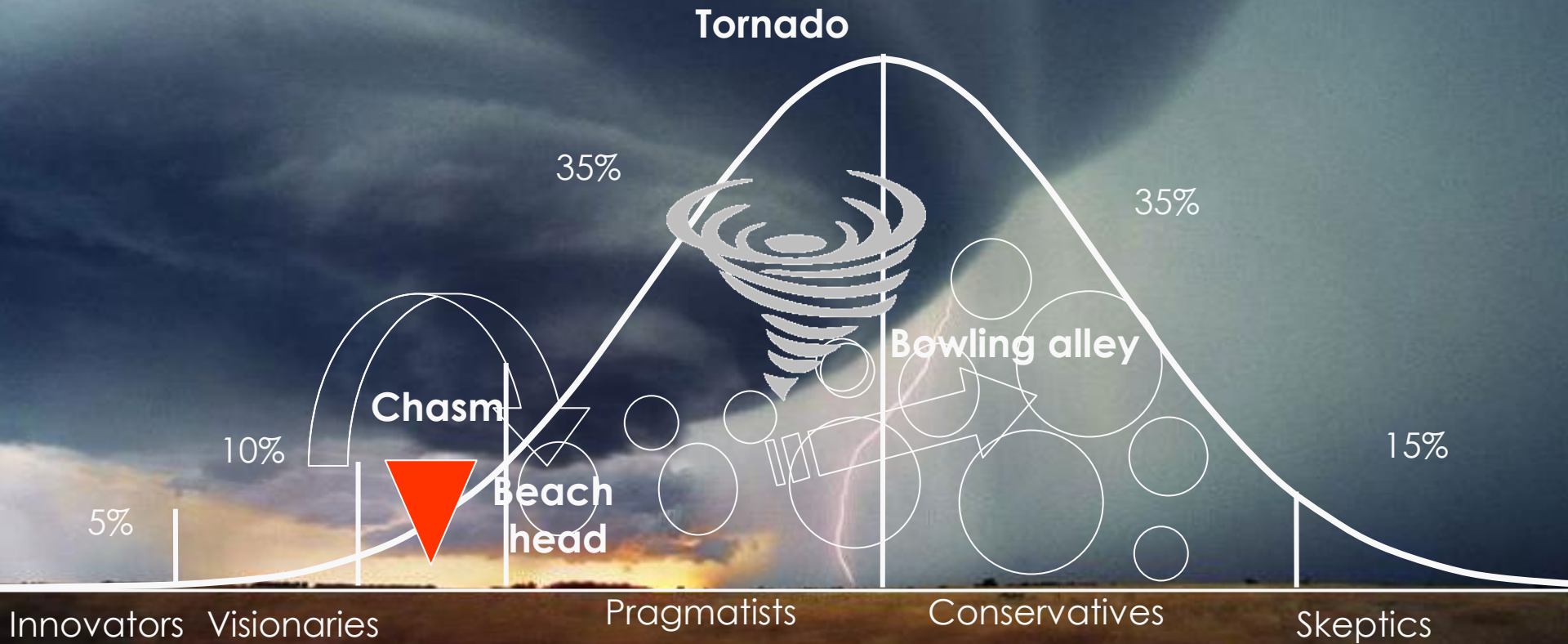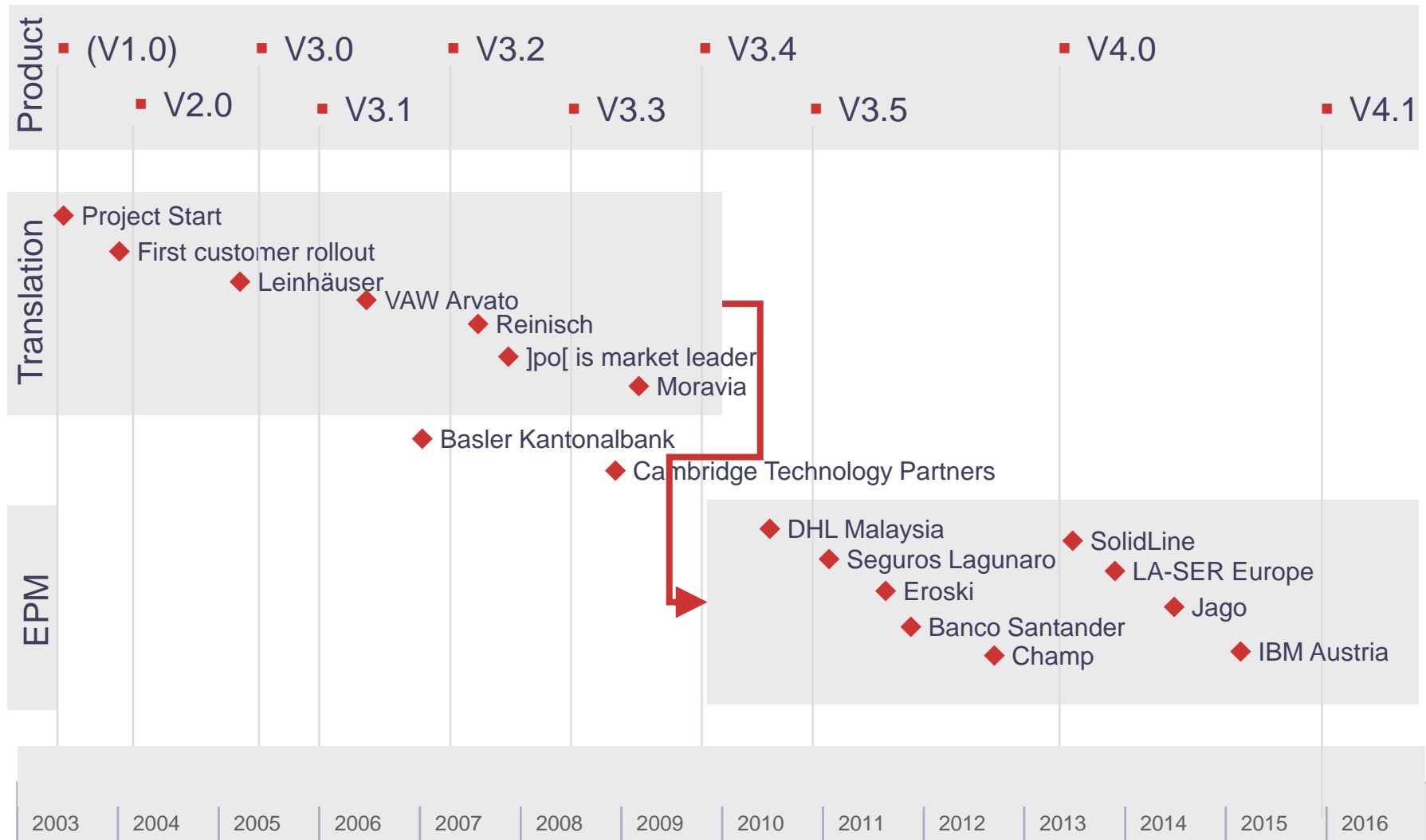- V3.1
- V3.2
- V3.3
- V3.4
- V3.5
- V4.0
- V4.1

Translation

- ◆ Project Start
- ◆ First customer rollout
- ◆ Leinhäuser
- ◆ VAW Arvato
- ◆ Reinisch
- ◆]po[is market leader
- ◆ Moravia
- ◆ Basler Kantonalbank
- ◆ Cambridge Technology Partners

EPM

- ◆ DHL Malaysia
- ◆ Seguros Lagunaro
- ◆ Eroski
- ◆ Banco Santander
- ◆ Champ
- ◆ SolidLine
- ◆ LA-SER Europe
- ◆ Jago
- ◆ IBM Austria

2003 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016
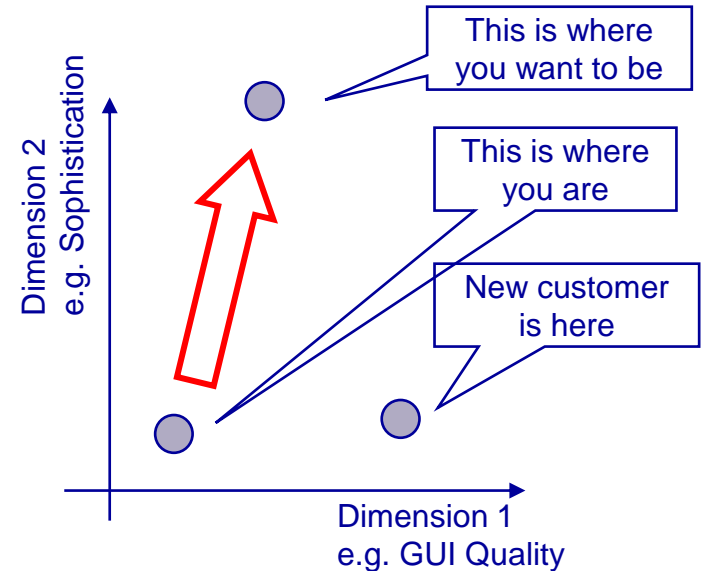
Zero Sales

- Wait until your customers approach you
  - *Some* customers actually research the market
  - Provide 10x better value than any competitor and/or specialize on a small niche
  - Maintain basic Google visibility
- Select the customers you want to serve
  - Strategic alignment of needs
  - Matching "customer sophistication"
  - Did download and check the open-source product
  - Have money
- Actual Sales activities
  - Product Demo
  - Reference customers
  - Quoting
  - ...
- Spread the word about your customers
  - Include viral elements in software
  - Write Success Stories (see HowTo)
  - Leverage open-source Marketing & PR

- Zero Sales in Open-Source can work because:
  - Creation costs are 10x – 100x cheaper compared with closed-source
  - You know that this is not cheesy, because of OSS dynamics
  - You know you'll have to put the screws yourself :-)
  - Not all Open-Source products may meet your quality criteria, though.

This is where you want to be

This is where you are

New customer is here

Dimension 2 e.g. Sophistication

Dimension 1 e.g. GUI Quality

What are you going to do?

Project vs. Product

	Project	Product
Customers	1	6.000
Business Objectives	Get stuff done	Do things right
Who takes investment decision?	The customer. We just execute	You yourself. Learn how to write off money.
Efficiency measure	Finish on time and budget	Time to market, differential functionality, low support
Code quality	Medium-low	High
How to deal with variability	Customize	Configure
Recommended Organization	Separate project team	Separate product team
Requirement Gathering	Requirement engineering with 1 customer	Understand market/business needs, check competition, listen to 6.000 customers, conferenes, biz textbook, ...

Zero Support

- Every bug will jump at you at some moment.
- 80% of our support cases come from specific (non-reused) code
- Whenever you add something to your system, consider the cost of maintenance
- Keep your Core clean

- Specific vs. re-used code
  - Maximize pieces of code used multiple times
  - Reduce size of code used only once per feature

- Customer customization requests
  - Make sure your business model doesn't need them…
  - Make them so expensive, that they become profitable
  - "Tooth extraction": Understand the business reason _behind_ the requirements, so that you can re-formulate them according to your product strategy

- General tips
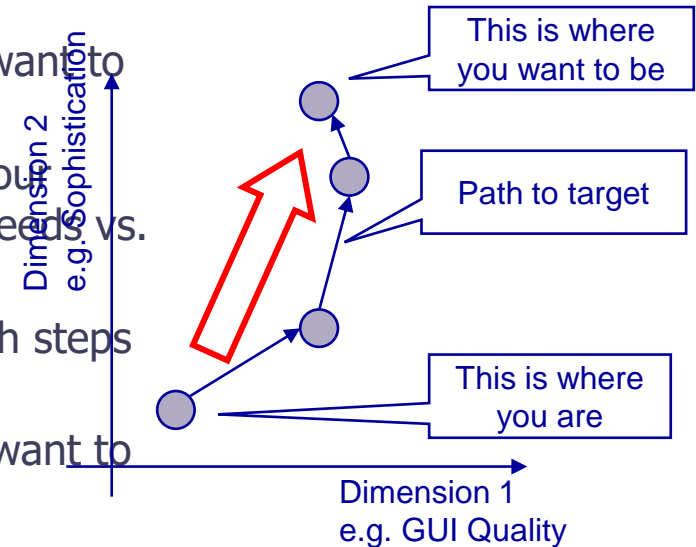 - Use configuration instead of customization
  - Use a powerful library

- Reduce "Entropy" (="disorder") in features and in general

Code Reuse & Efficiency
- Open-Source Model
- "Catapult Start"
- "Raw Diamonds"
- "MBA Codes"
- Mature and complete library
- "Sunflower Model"
- Highly productive functional language

Business Strategy
- "Bridging the Chasm"
- "Customer Self-Selection"
- "Product Management"
- "Zero Sales"
- "Zero Support"

Code Economics
- "Sunflower Model"

- Define medium and long-term goals "where you want to be" with your product
  - Define customers you want to serve
  - Define dimensions of your product vs. customer needs vs. competitors
  - Develop a roadmap with steps how to get there
  - Define what you *don't* want to do
- Follow the strategy, unless a customer *really* pays a lot of money (it may be time to revise the strategy then...)
- Move non-strategic stuff into containers (add-on modules), so that it doesn't pollute the core.

Dimension 2
e.g. Sophistication

Dimension 1
e.g. GUI Quality

This is where you want to be

Path to target

This is where you are

What are you going to do?