PEOPLE | SOFTWARE | HAPPINESS™

BUSINESS

LONG-TERM COMMITMENT

Long-term dedication, commitment and involvement focuses on success and a fruitful relationship. Go for the long-term and grow a profound and healthy, sustainable business and organization.

QUARTERLY CONTRACTS

A quarterly happiness check and temperature reading allows all parties to look backward and forward to assess the effectiveness and joy of working together.

The decisions are ones of allocating business resources and belong in upper management. However, all present can contribute, and should do so in a frank, honest, non-defensive and constructive way. It gives all stakeholders the opportunity to tune and extend the relationship.

The happiness check also gives the opportunity to loosen up or terminate the agreement if desired.

Finally, this quarterly recommitment meeting is a good kick-off for another fruitful development episode.

Software development continues as long as returns on investment make it viable. Consulting and coaching are no exception and will eventually be scaled down.

It is our intention to get you up to and self-supportive as soon as possible, leaving behind a thriving sharp and focused team that excels in its practice.

VALUABLE STORIES

Requirements, features and crystal clear acceptance criteria are captured as Valuable Stories—user scenarios as discrete units of value creation mapped to specific software features.

Stories are the basic elements incremental funding during agile development. Completing Valuable Stories earn Royalty Points.

INCREMENTAL FUNDING

Incremental funding quantifies the financial benefits—value—of individual features of a software development project and optimizing the delivery sequence for maximum financial impact, while reducing risk.

SUCCESS BRINGS BONUS

A fixed price project has the serious potential of losing out on quality: you're running out of time (and resources), yet the project must be delivered on the deadline. On the other hand, on a time and material basis, development can be endless, delivering too little, too late, while still having to pay up.

An excellent way to align the interests of all stakeholders is to provide a bonus for timely completion of the project. The control given by agile planning makes it very likely that the development team will be able to collect.

The evil twin of the completion bonus is the late penalty. Again, agile planning gives the development team an advantage when agreeing to a late penalty. The team can be quite sure that they will complete the system on time, so they are unlikely to have to pay.

You can even go a step beyond that and make the whole a community-owned project.

RESULTS

A SOLUTION THAT YOU WANT TO BUY

The end result is an excellent (software) solution that people want to buy. A solution that works as advertised. A solution that is cost-effective, easy to use, highly evolvable and low in maintenance and customer care. A solution that helps clients succeed and results in a profitable enterprise for both client and supplier. It's fast, beautiful, menu-free, dead simple to operate, and a pleasure to use.

It's beautiful and breakthrough. Unprecedented. People lust for it.

Besides an excellent solution filling in a key need in the market, it also generates concrete results like a common vocabulary, pattern languages for organization, community, architecture, process and design, a comprehensive user manual and realistic estimates for future versions.

QUALITY WITHOUT A NAME

The richness of the common vocabulary through the extensive use of patterns and pattern languages, the comprehensive user manual, the personas, the smooth agile processes, the focus on a solid architecture, the predictable rhythm of sprints, development episodes and the quarterly releases all add up to a comfortable and highly evolvable team, process and product with a Quality Without A Name and a Great Place to Work.

EVOLVABILITY IS KEY

For extremely profitable price/quality ratios, aim to maximize evolvability for all stakeholders by:

- minimize the efforts to develop, release and maintain your product;
- maximize scalability and evolvability;
- maximize innovation to happen elsewhere;
- maximize adopting & setting open standards for quality, processes and technology.

The evolvability quality requirement transcends and includes all other "ilities" like performance, scalability, maintainability, reliability, resilience, and security.

USER MANUAL

The user manual is a clear and comprehensive part of the product and allows project team, vendor, press and users to confirm that the product works as advertised, creating the right buzz in the market.

INTELLIGENCE

CONWAY'S LAW

Knit, weave and nourish the close relationship that exists between the structure of an organization and the artifacts that it builds.

LEARNING ORGANIZATION

Turn tacit knowledge into explicit knowledge through Excellence Guides and a coherent set of Pattern Languages. Provide room for feedback, retrospection and temperature readings.

COMMON VOCABULARY

The on-line common vocabulary that emerges during development is captured for both the project team and the market and facilitates clear communication between all stakeholders.

A shared language coheres the management team, development team and users of the product.

The extensive use of organizational, architecture and design pattern languages add to the rich and comprehensive vocabulary, easing communication and reducing failures and errors while capturing, consolidating and evolving a collective intellect. Getting better is constantly getting better.

ORGANIZATIONAL PATTERNS

Organizational patterns are applied to enable a piecemeal growth of team and product. Rapid feedback and temperature readings lead to the emergence of new organizational patterns. The organizational patterns also foster the use of a common vocabulary, allowing late joiners to quickly get up to speed and contribute to the product's success.

COMMUNITY PATTERNS

Community patterns help to create the initial communities and to evolve them over time. Use the community patterns to grow both the end-user community and developer community.

The community patterns focus on five key areas: process, the marketplace, fairness, product development, and quality and stability.

Healthy and smart communities turn companies into winners.

ARCHITECTURE PATTERNS

Architecture patterns are essential to guarantee good scalability, performance, security, reliability, availability, evolvability, etc. Applying and evolving the architectural patterns is key to maximizing return on investment and profit, and minimizing marketing, sales and customer care efforts.

DESIGN PATTERNS

Design patterns are just like architecture patterns, yet on a smaller scale. Design patterns found in the industry can be applied, and new ones will emerge and documented, speeding up development and transforming tacit knowledge for a few into explicit knowledge for others.

PROJECT MANAGEMENT PATTERNS

Project management patterns help the development team to enter a state of flow, where production and quality come effortlessly. They have to do with the work of the organization and the manner in which that work is structured. It focuses on schedule, process, tasks, and in particular the structures needed to support good work progress.

PIECEMEAL GROWTH

Piecemeal growth helps you grow the organization and its processes together. It is reminiscent of concurrent engineering approaches that grow the process and product together.

ARCHITECTURE EVERYWHERE

Deep insights on organizational structuring in light of growing insight into the system architecture.

EXCELLENCE GUIDES

Excellence Guides collect and document the do's and don'ts for all team members. The principal format is "You must...", "You must not...", "You should...", and "You should not...".

For example: You must Develop customer scenarios before the Design phase. These rules of the game capture the knowledge and practices and help later joiners get up to speed quickly.

Create and evolve Excellence Guides for Usability Engineering, Development, Program Management, Marketing, Product Design, Localization, Content Publishing, and Testing.

PREDICTABLE VELOCITY

After each sprint, velocity becomes more and more predictable. Both individual and team velocity exhibit better predictability as the project unfolds. Changes in team or technology have buffered impact on velocity, allowing for comfortable planning and releases, and realistic estimates.

QUARTERLY RELEASE RHYTHM

The quarterly release rhythm brings calmness to both the market and the development team. These 13-week development episodes provides clear and predictable deadlines for all stakeholders.

Suit these 13 week development episodes to fit what feels natural to you:

- 12 sprints of 1 week;
- 6 sprints of 2 weeks;
- 3 sprints of 4 weeks; or
- 4 sprints of 3 weeks.

Each DEVELOPMENT EPISODE with one "week out of time" to celebrate the previous one and get ready for the next one.

Each release is named after the releases season; for example the product version 2008 Spring release. Optionally, use automatic software updates to ease maintenance at client installations.

A GREAT PLACE TO WORK

Shared values and open communication help create a vibrant and healthy environment where individuals can exceed their goals while having fun.

INNOVATION

USER COMMUNITY

Grow a vibrant and active user community. Turn some of them into evangelists and ambassadors. And listen. Listen proactively to what they are saying. Design short feedback loops so that they feel heard. Breed fanship and consider crowd-sourcing.

CROWD-SOURCING

Connecting customers into a collective intelligence and encouraging them to talk to each other, to form affinity groups and hobby tribes, will breed smarter and more loyal customers quicker while creating smarter products and services.

Therefore:

- Make customers as smart as you are.
- Connect customers to customers
- Choose technology that connects.
- Imagine your customers as employees.

COMMUNITY-OWNED

Everyone who joins the community has the right to own a share in the co-op. The co-op is established for the advancement of technological entrepreneurship and for the benefit of its members. The whole idea is to tie productivity to reward.

The co-op's profit is your profit. The size of your piece of the pie depends on your level of participation in the community.

As a co-op member, you get a vote on all matters respecting the co-op and are entitled to patronage dividends in the form of cash and/or member shares as determined by the governing board in response to member participation within the community.

Your work, visibility, activity and contribution collects Royalty Points as payment for your work, and Glory Points for participation—both for as long as your work drives profit. The more results you contribute and the more you participate, the more dividends you'll earn. The pay-out is simple: Pay-out = Your Royalty Points / Total Royalty Points * Profit.

Directors (co-op community members elected by people like you) ensure equity and profits are shared.

THRIVING OPEN SOURCE

Most of the smart people do not work for you. So innovation happens elsewhere. Tap into this vast source of innovation and creativity by making everything you do open source—when not in conflict with your competitive position. Adopt a dual licensing strategy to create a viable and healthy commercial open source effort.

Make sure you set up an appropriate developer community process. Mix and match open source and crowd-sourcing into a flourishing community-owned enterprise.

Make giving and sharing your second nature. Sharing is multiplying.

CONSCIOUSNESS

GREEN & CONSCIOUS COMPUTING

Put Planet before People, Profit and Products, since in the end, it will benefit all, creating a Whole Earth.

Proactive study and practice of using computing resources efficiently. Take into account the so-called triple bottom line of economic viability, social responsibility, and environmental impact.

Select vendors and partners with a proven track record for green computing or "conscious computing". Google, Apple Computer, and Sun Microsystems are leading various efforts in conscious computing.

AardRock

Vision: **Agile Business**	
Author: Martien van Steenbergen	
Version: 2	Date: Winter 2007

PEOPLE | SOFTWARE | HAPPINESS™

PRINCIPLES

PRINCIPLES OF PRACTICE

1. **PLANET**—Focus attention on resilience and sustainability for our planet and our company, and in that order, profitably, consciously.
2. **FRIENDS**—Make friends while having fun.
3. **WEALTH**—Intensely focus on building wealth for our clients by symbiosis.
4. **DIVERSITY**—Work to ensure diversity of people, communities, and technologies.
5. **INTEROPERABLE**—Work to ensure that technologies used as part of the system are fully interoperable with one another, on the highest semantical and spiritual level.
6. **INNOVATIVE**—Embrace methods, innovations, technologies and solutions that are unbelievably creative and disruptively innovative.
7. **RESPECT**—Be open in vigorous debate and dialogue; with deep respect for the individual and the team.
8. **WISDOM**—Appreciate knowledge and wisdom.
9. **ENLIGHTENMENT**—Encourage the enlightenment and development of team members and our clients.
10. **OPEN**—Freely and fully exchange information relevant to our purpose, mission, vision and principles unless it violates confidentiality or materially diminishes our competitive position.
11. **COMPASSIONATE**—Resolve conflict without resort to economic, physical or other violence or intimidation.

PRINCIPLES OF ORGANIZATION

1. **OPEN**—Be open to owning membership by any individual or institution subscribing to the purpose, mission, vision and principles in conducting activities that resonate with and fuel our practice.
2. **EXPRESSION**—Have the right to self-organize at any time, on any scale, in any form or around any activity consistent with the purpose, mission, vision and principles.
3. **FAIRNESS**—Work to ensure that no member obtains an intrinsic unfair advantage in the system.
4. **COMMONS**—Work to ensure that voting rights and membership (fees) are derived from a common formula based on each members' contribution to the system.
5. **INVOLVED**—Conduct deliberations and make decisions by bodies and methods that reasonably represent all relevant and affected parties and are dominated by none.
6. **EMPOWERED**—Vest authority, perform functions and use resources in the smallest or most local part that includes all relevant and affected parties.
7. **EDUCE**—Educe rather than compel behavior to the maximum possible degree.

PRACTICES

RISK MANAGEMENT

RISK—Next version of Windows ships too late.
OWNER—Leadership team.
IMPACT—Will not be able to ship new version on time.
PROBABILITY—High.
RISK COST— Lost revenue; closing window of opportunity; reputational damage.
MITIGATION—1) User current stable version; 2) Develop abstraction layer.
MITIGATION COST—Low: One month of delay but still one month before planned release date.

The RISK LIST is managed on a daily basis. Mitigation decisions are made during the weekly planning game. Everyone can contribute to the RISK LIST. The Project Manager owns the RISK LIST. A Risk has clear ownership: either the development or the management team.

A management team-owned risk is beyond the development team's ability to change its probability of occurrence. A management team-owned mitigation plan is one that the development team will not undertake (but may still track). Impact (low , medium, high) and Probability (low, medium, high) are indicators for the risk's Exposure.

ORGANIZATIONAL PATTERN LANGUAGE

We will use organizational patterns, which are about people, in order to do effective (software) development in a healthy, thriving organization.

The team leader and the management team support shaping the organization and help individuals and the organization to become effective, instill a UNITY OF PURPOSE and apply selected patterns.

TEMPERATURE READING

APPRECIATION—"I appreciate you because/for..."
NEW INFORMATION—"Have I got news for you!"
PUZZLES— "I'm confused about..."
COMPLAINTS AND RECOMMENDATIONS— "My complaint is... And here's how I think we can resolve it..."
HOPES & WISHES— "My wish is that you..."

Software projects can be chaotic and stressful. Conduct TEMPERATURE READINGS as a simple yet powerful method to help the team achieve, maintain, or regain a positive mindset. It helps the team reduce tensions, solidify connections, and surface important information, ideas, and feelings—enabling team members to interact more constructively and productively. It is a superb technique to use at project milestones or during team meetings for teams that work together under demanding or deadline-driven circumstances.

LIVELY PERSONAS

LIVELY PERSONAS are a tool to use when designing your product. Closely related to customer scenarios or stories, LIVELY PERSONAS define who you're designing your product for. Using LIVELY PERSONAS helps to create a common language for everyone designing the product and helps to avoid the ambiguous term "users".

Ideally, personas and scenarios are seamlessly aligned. To achieve that ideal, make sure that a relevant persona exists for each user who you target, and then use only that persona in a consistent and relevant way. To apply the full power of personas, understand the nature of the persona you're using in the scenario.

USER MANUAL

The USER MANUAL, written and maintained by the MERCENARY ANALYST, collects all stories in a comprehensible and cohesive way. Elegant, consistent, and clean. Both the code and the manual must do and say the same. It's considered a bug if they're not.

The MERCENARY ANALYST's primary motivation is to get the software—not the documentation—out faster! MERCENARY ANALYST comes from the "hired gun" quality a MERCENARY ANALYST might have; rides into town, gets the early stuff documented, kisses his horse, saddles up his girl, and rides off into the sunset.

RAPID FEEDBACK & REFLECTION

Key questions to ask on a regular basis are:

- What did we do well?
- What have we learned?
- What can we do better?
- What still puzzles us?

The development process will change over time. A project that begins using an adaptive process will have a different process a year later. Over time, the team will find what works for them and alter the process to fit. This self-adaptive process helps the team and the whole organization getting better in getting better.

DAILY STAND UP MEETING OR SCRUM

Everyday the team holds a short fifteen minute meeting, called a SCRUM, where the team runs through what it will do in the next day. In particular, the team will surface the risks to success that management needs to resolve. The team also reports on what has been done so management can keep track.

TEST-DRIVEN DEVELOPMENT

USER FEEDBACK TESTS—Guides real users through a session with real software and the development and management team in another room watching. Solves the "Don't Make Me Think!" problem. Instills common sense into the product.
FUNCTIONAL TESTS—Exercise the system from the user's point of view and define progress to date.
UNIT TESTS—Exercise fine-grained components within the system and help developers think through the structure of their code.
INFRASTRUCTURE TESTS—Ensure that the application subsystems are communicating properly and deployment runs smoothly.

CONTINUOUS INTEGRATION

CONTINUOUS INTEGRATION solves many of the issues that plague large-scale development: avoids nasty surprises as the project nears delivery: there is always a working application the grows organically; good visibility into progress; alignment of potentially distributed teams. Requires commitment and some ingenuity and reduces stress level on a project.

AUTOMATES REPETITIVE ACTIVITIES—Integrates so often that it becomes easy. Automated deployment scripts reduces costs and errors, eliminates ambiguities and forces clear boundaries. Uses computer rather than human power.
LITTLE CHANGES HAPPEN OFTEN—Build continuously to find and repair failures when only a few things have changed. Gives fine-grained control over the history of your application.

MANIFESTO

PASSION MANIFESTO

CREATING GOOD SOFTWARE NEEDS PASSION—passionate developers, who want to create something wonderful; passionate customers, who want something that will enrich their lives. We don't just choose people who are technically gifted.
PASSION THRIVES ON FREEDOM—freedom to choose what we work on, and who we work with. We don't say "yes" to every customer.
PASSION THRIVES ON INVOLVEMENT—and to be truly involved you need to understand the context, including the economic context. All our financials, including salaries, are open to all our staff.
MAINTAINING PASSION MEANS ACCEPTING LIMITS—in particular, limits to growth. We would rather turn work away because we don't have anyone available than compromise by hiring the wrong people.
YOU SHOULD DO EVERYTHING YOU CAN TO GET THE RIGHT PEOPLE ON THE BUS—this is one of the few reasons to grow, because collaborations between the right people will create things no individual would imagine
SUCCESS MEANS SUCCESS FOR EVERYONE—our economic metric is profit per employee, not overall profit or revenue.

AGILE MANIFESTO

We are uncovering better ways of developing organizations, software, and solutions by doing it and helping others do it.

Through this work we have come to value:

- Individuals and interactions over processes and tools.
- Working software over comprehensive documentation.
- Customer collaboration over contract negotiation.
- Embracing to change over following a plan.

That is, while there is value in the items on the right, we value the items on the left more.

RIGHTS

CLIENT RIGHTS

As a client and funder of the project, you have the right to:

1. Plan on a large scale with investments and options.
   Set development priorities weekly.
2. See progress in the form of a working system at the end of the first week, and to see a little more functionality every week thereafter.
3. Updates of the schedule, good or bad, as soon as the information is available.
4. Change your mind without paying exorbitant costs.
5. Stop the project at any time and still own a usable system on par with the investments made up to date.

DEVELOPER RIGHTS

As a developer, you have the right to:

1. Always know what needs to be produced, with clear requirements, acceptance criteria, and priorities.
2. Estimate work and have those estimates respected by the rest of the team.
3. Honestly report progress with impunity.
4. Produce high-quality work at all times.
5. Know what is most important to work on next.
6. Ask business-oriented questions whenever they arise and even if they are uncomfortable ones.
7. Facilitate a self-selecting team.
8. Engage in joyful, exciting, challenging and productive work.

PROJECT LEADER RIGHTS

As a project lead or manager, you have the right to:

1. An overall estimate of investments and results, recognizing that reality will be different.
2. Move people between projects without paying exorbitant costs.
3. Regular updates of progress resonating with the rhythm of the business, and to help the customer set overall priorities.
4. Focus on personal development of team members.
5. Cancel the project and be left with a working system reflecting the investment to date.

ARCHITECTURE BOARD

The structure of the organization, processes, teams, infrastructure, and software is the pivotal responsibility of the Architecture Board—providing just enough structures, standards and conventions for creative tension and direction and sufficient freedom for self-expression of individuals and teams. It is, in fact, the principal force behind a creative, innovative and chaordic enterprise experiencing frequent states of flow.

The Architecture Board also puts up antennas to pick up any new developments—technological, organizational, economical, social, cultural—and integrates these into the relevant areas of the whole.

Last but not least, the Architecture Board captures and documents good practices, lessons learned, and experiences into Pattern Languages and Excellence Guides. It actively shares these across and beyond the community, thereby creating an agile life-long learning organization. It embodies the Community of Practice.

The Architecture Board has a wide focus and sufficient depth when needed. It is accountable for what is important besides the code and it rather solves problems and craves solutions by ignoring details. It keeps the organization as a big team together, maintaining architectural integrity, communicating and leading. And it buffers the techies versus the non-techies. Wholeness.

BIG VISIBLE CHARTS

Track trends, history, progress, acceptance tests and sprint burn-down, or sensitive subjects use a simple BIG VISIBLE CHART on the wall and bring important information to the attention of the team, the client, and everyone else who passes through the area.

Provide important information, even politically sensitive information, without getting personalities involved or hurting feelings.

BIGGER IS BETTER—A chart drawn on a sheet of flip-chart paper can be seen from across the room. It draws the eye, pulls you over to take a closer look.
CASUAL IS BETTER—Consider simple charts over fancy, advanced, professionally designed charts. Draw your chart with whiteboard markers on a tablet of flip chart paper. You can update it in seconds every day. If the chart needs to be a bit different, rip it up and draw a new one.

You'll save time, have more fun, and the charts will be more personal and less mechanical.

AardRock

Vision: **Agile Business**

Author: Martien van Steenbergen

Version: 2 | Date: Winter 2007