

MADS

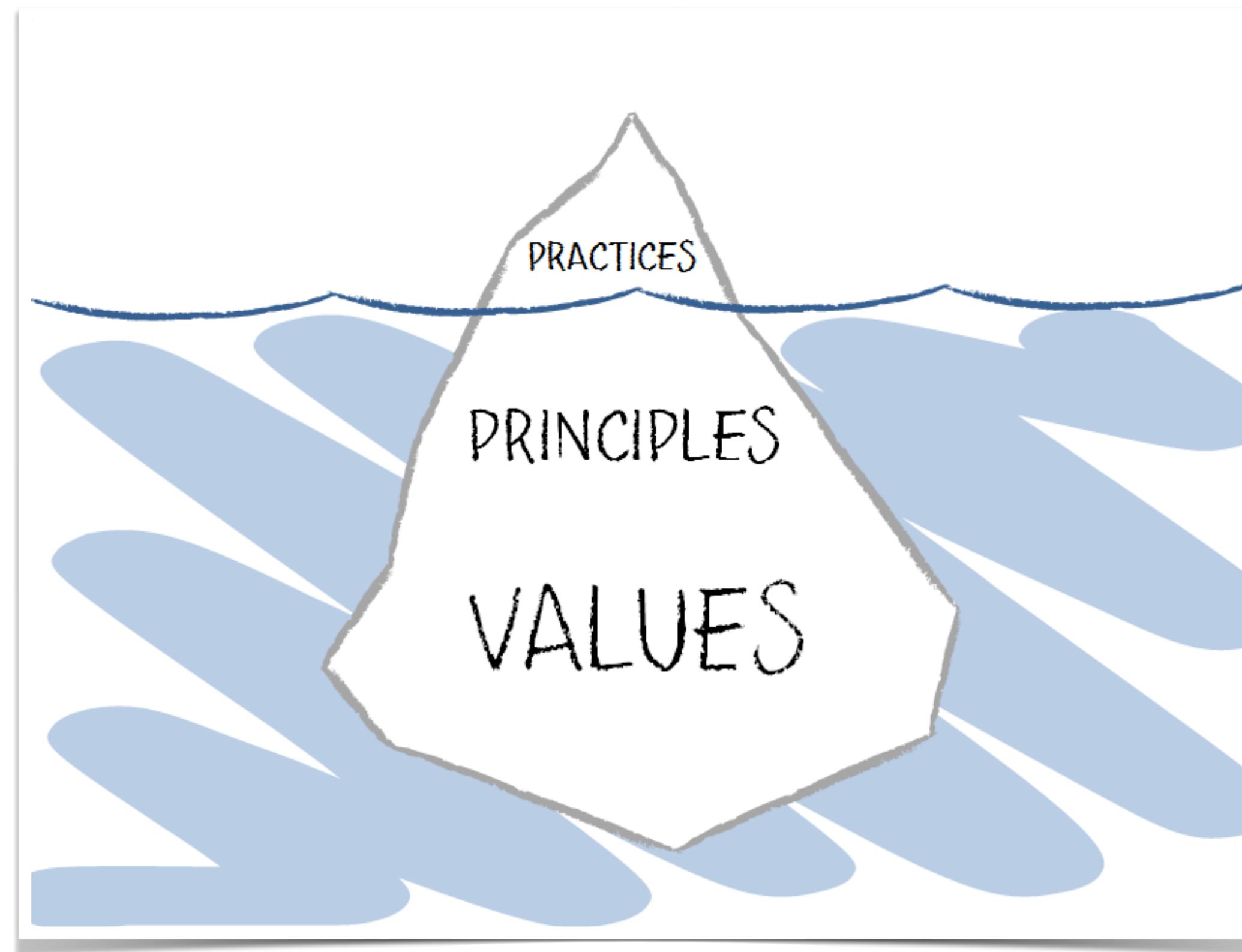
S4: El Manifiesto Ágil

Índice

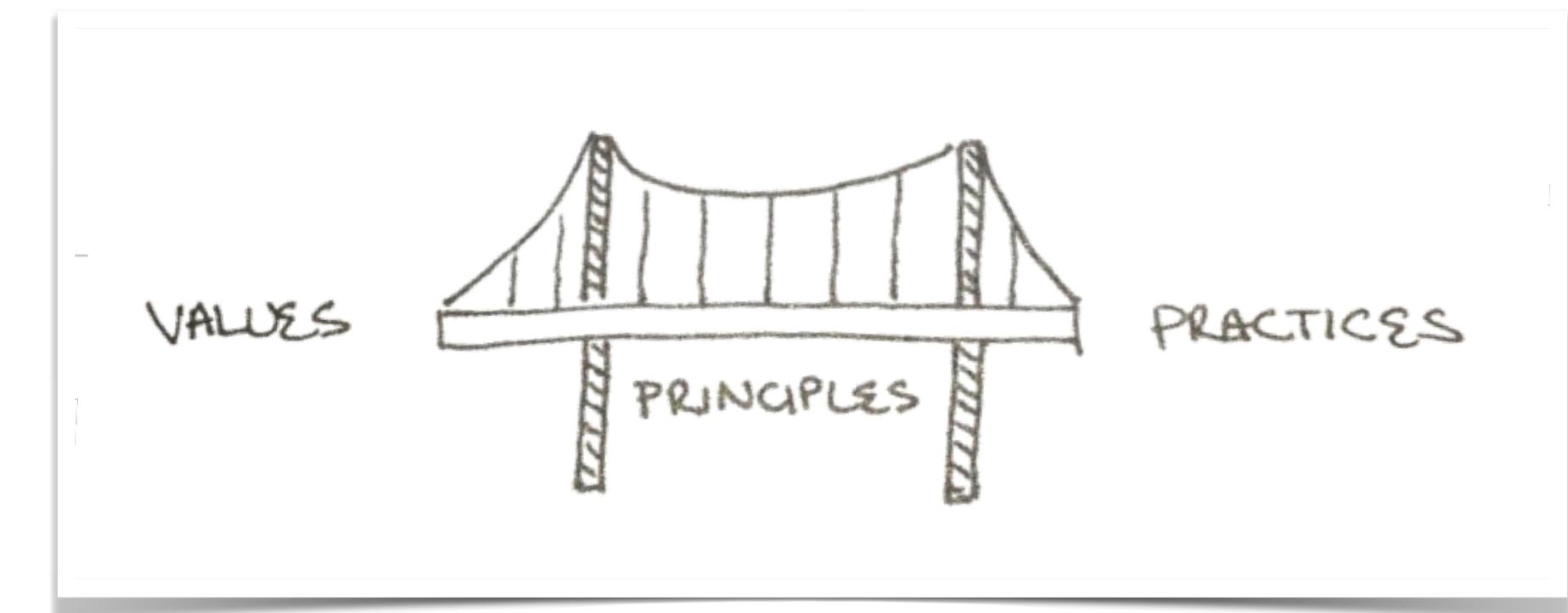
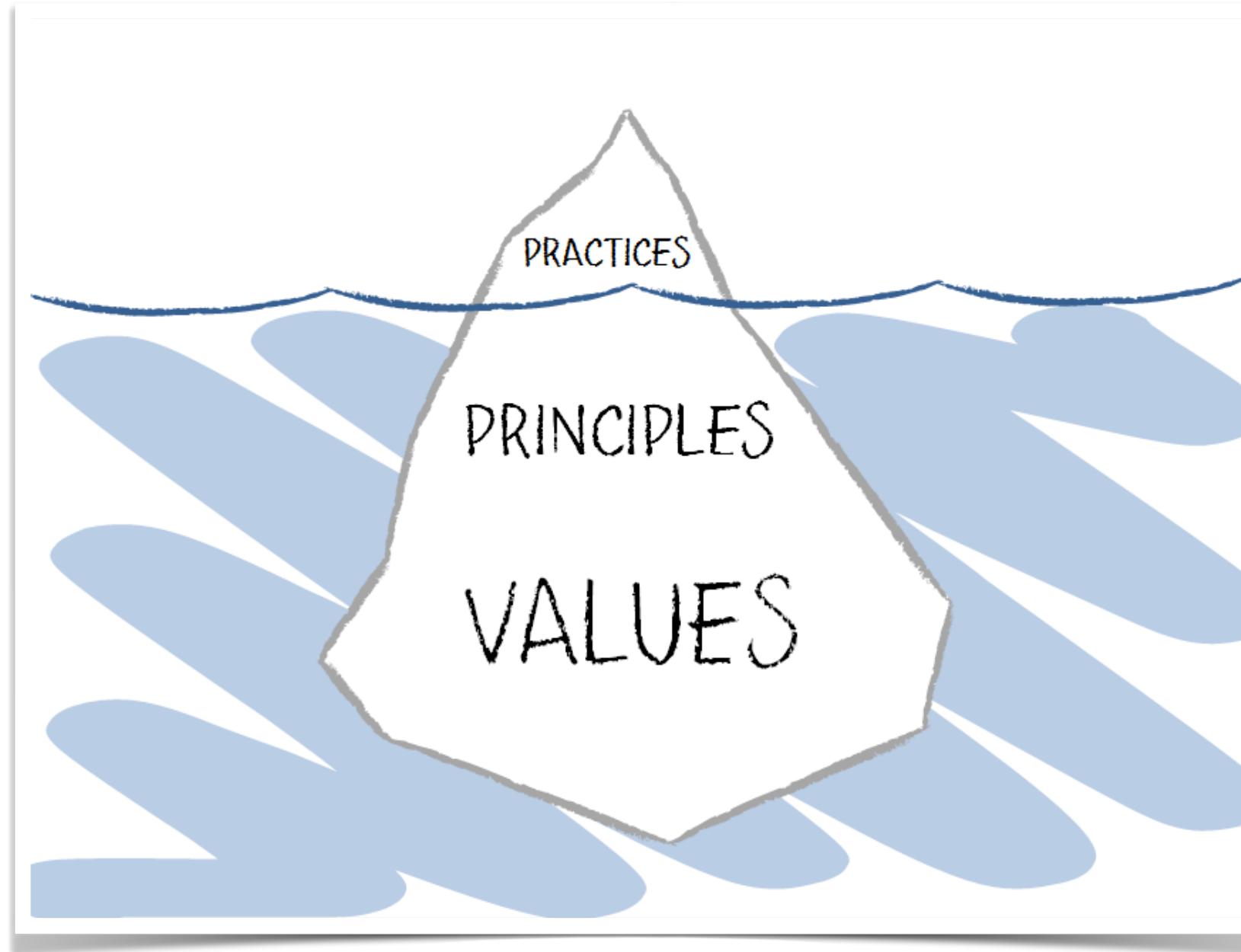
1. Actividad: nuestro manifiesto para mejorar el desarrollo de software
2. Tendencias previas al manifiesto ágil
3. El Manifiesto Ágil

1. Actividad: nuestro manifiesto para desarrollar mejor el software

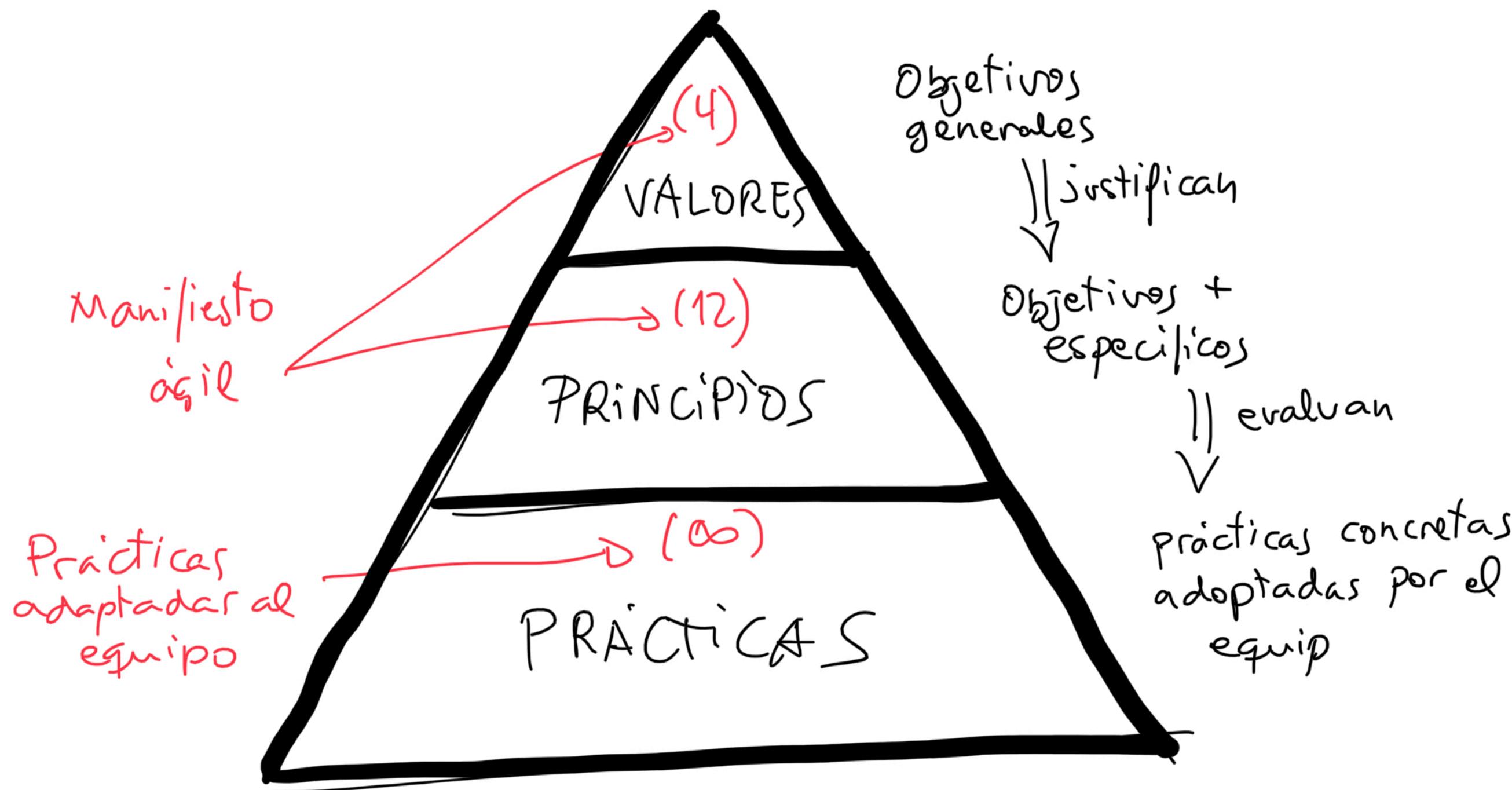
Valores, principios y prácticas



Valores, principios y prácticas



Valores, principios y prácticas



Actividad en mentimeter

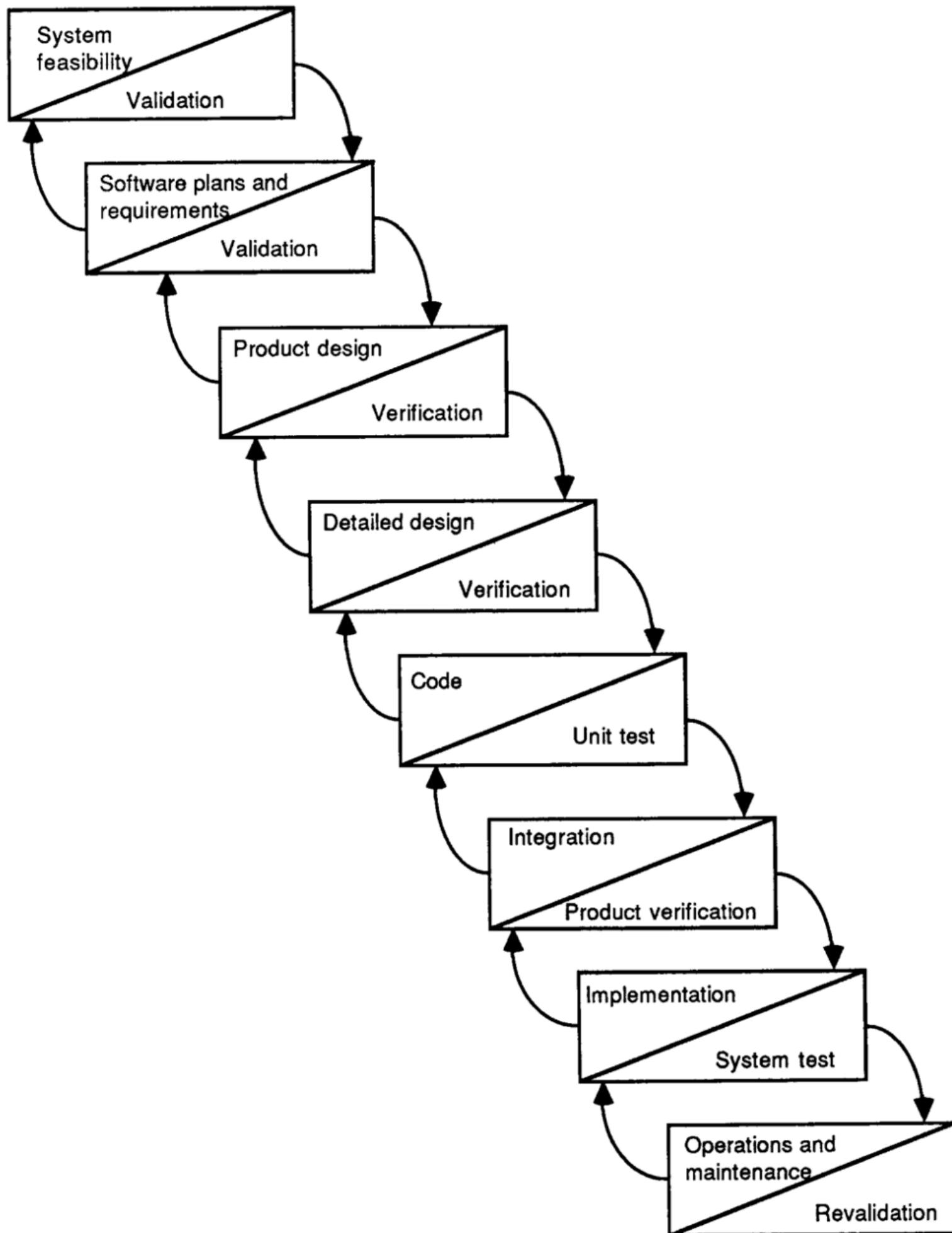
- Vamos a hacer **nuestro propio manifiesto** de desarrollo de software.
- Grupos de 3 personas. Durante 5 minutos cada grupo escribe 3 objetivos (principios) que deberíamos de cumplir para desarrollar software con éxito.

“Tenemos que ...”

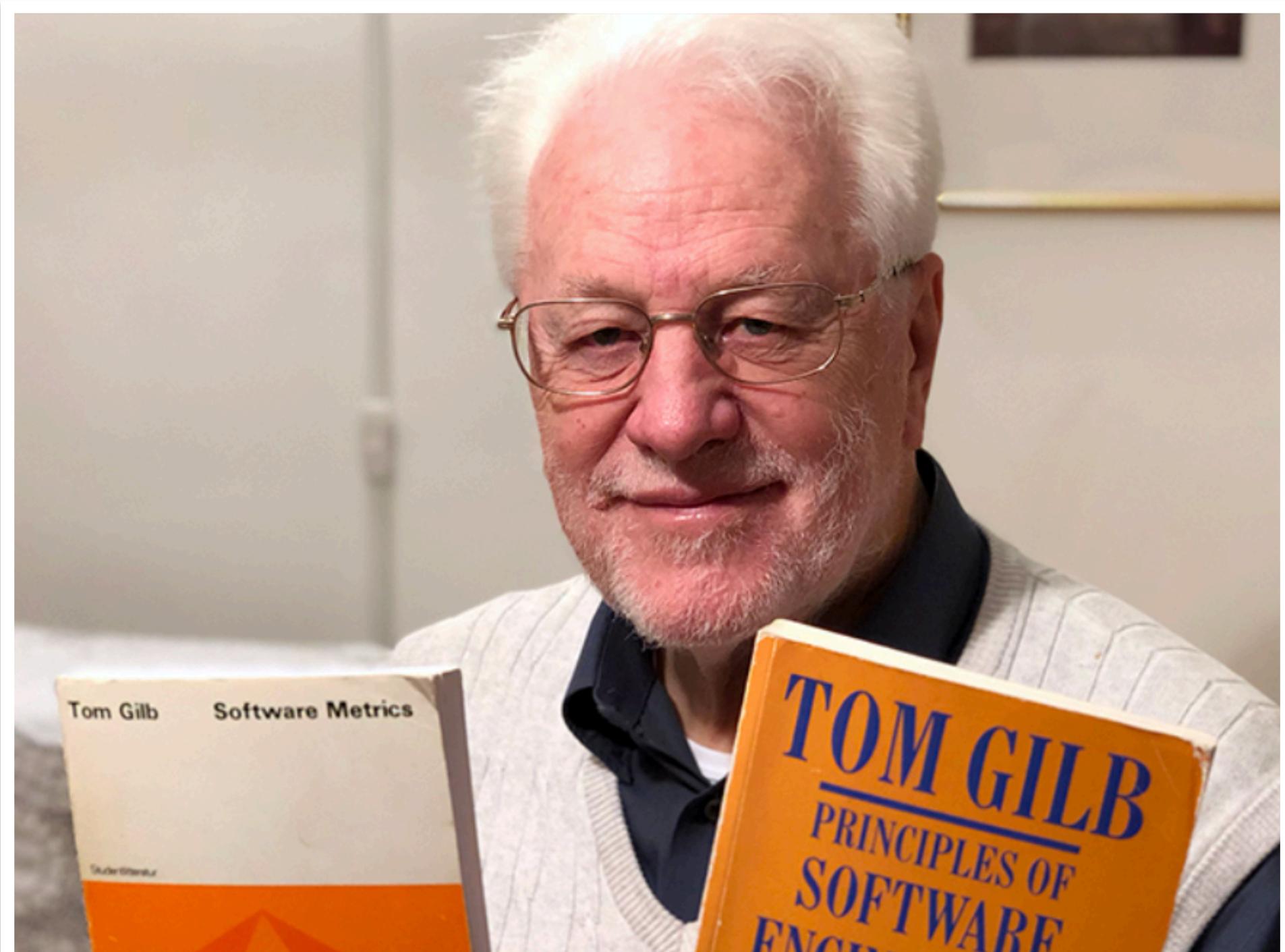
- Se ponen en común escribiéndolos en mentimeter.
- Filtraremos todos los principios y nos quedaremos con 8/10.
- Después intentaremos definir los 2/3 objetivos generales (valores) que justifican estos principios.

2. Tendencias previas al manifiesto ágil

1970



1976



“Evolution” is a technique for producing the appearance of stability. A complex system will be most successful if it is implemented in small steps and if each step has a clear measure of successful achievement as well as a “retreat” possibility to a previous successful step upon failure. You have the opportunity of receiving some feedback from the real world before throwing in all resources intended for a system, and you can correct possible design errors.

Tom Gilb (Software Metrics, 1976)

1980

DISPLAY CUSTOMER INFORMATION

Recd # B10093

Credit Limit:\$ 0 Finance Charge? Y Area: Sort Codes: 8

BILLING

Name: A CLEAN WELL LIGHTED PLACE FOR
Address: 601 VAN NESS AVENUE
City: SAN FRANCISCO
State: CA
Zip: 94102
Country: U.S.A

SHIPPING

Name: A CLEAN WELL LIGHTED PLACE FOR
Address: 601 VAN NESS AVENUE
City: SAN FRANCISCO
State: CA
Zip: 94102
Country:

Phone:

Phone:

Enter ↑ to skip back, ↓ to skip forward, or <ESC> to exit



1980



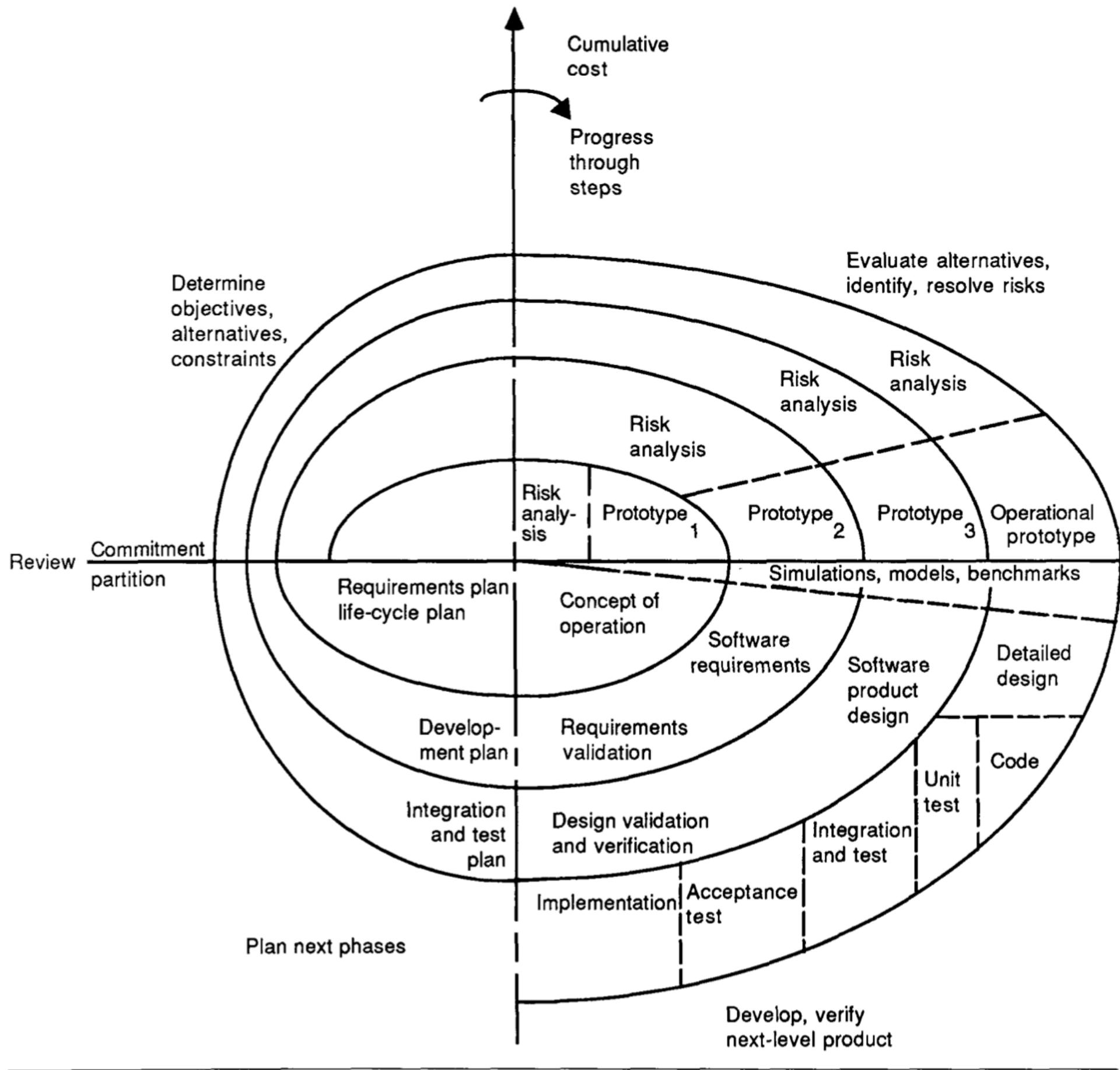
With object-oriented design we never encounter a "big-bang" event of system integration. Instead, the development process results in the incremental production of a series of prototypes, which eventually evolve into the final implementation.

Object Pascal



1985

Bohem Spiral model

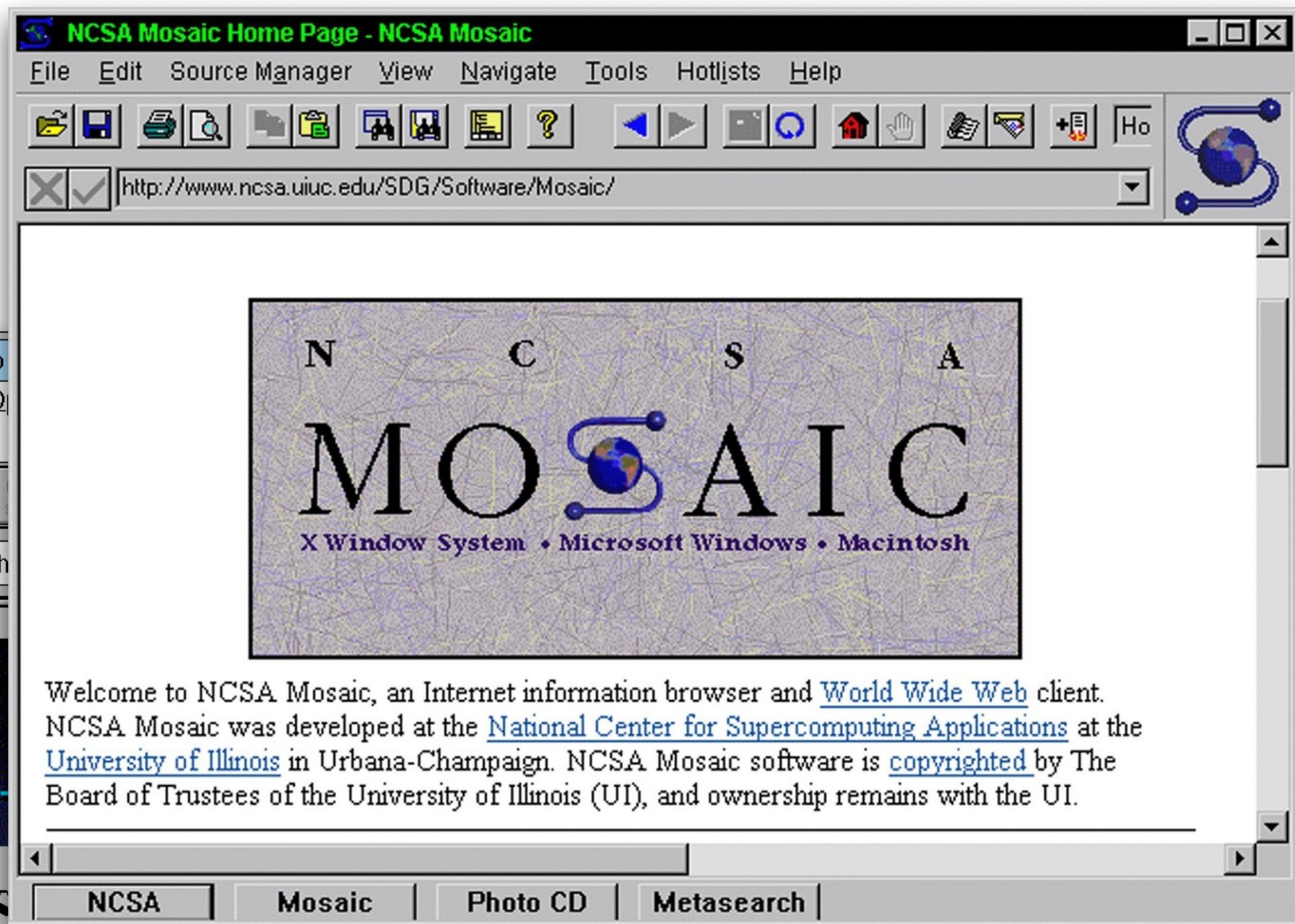


1993

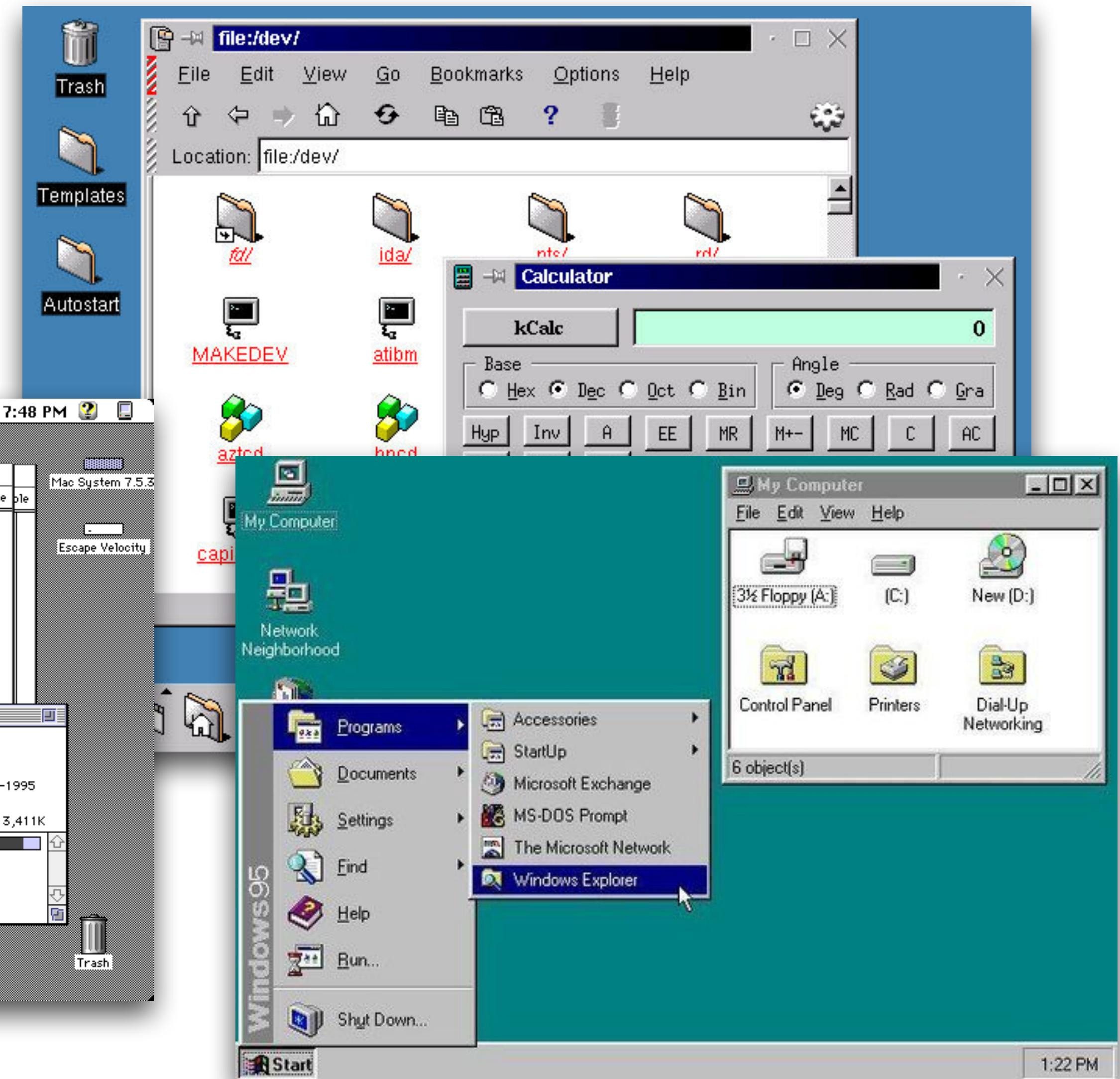
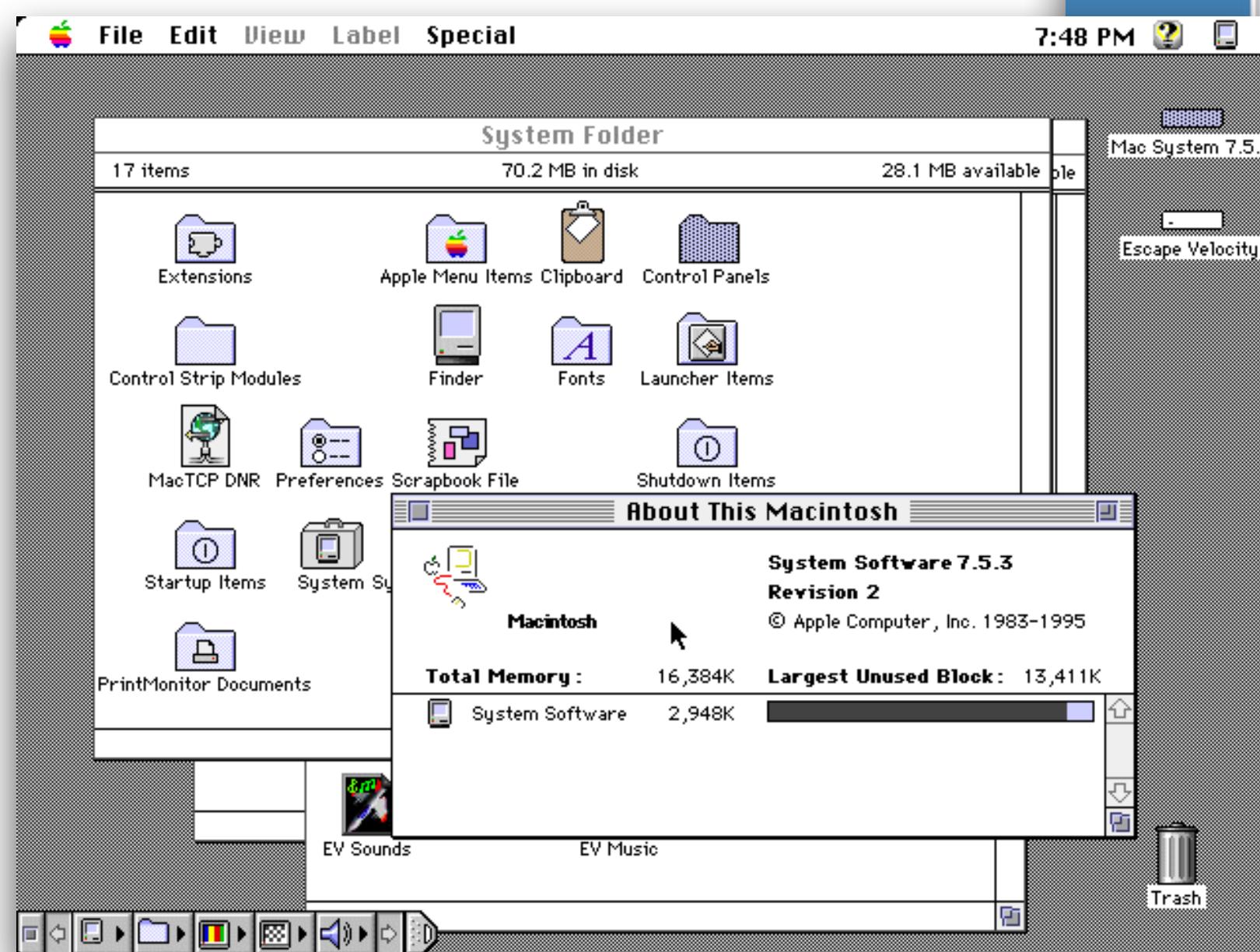


How to Create WWW Services

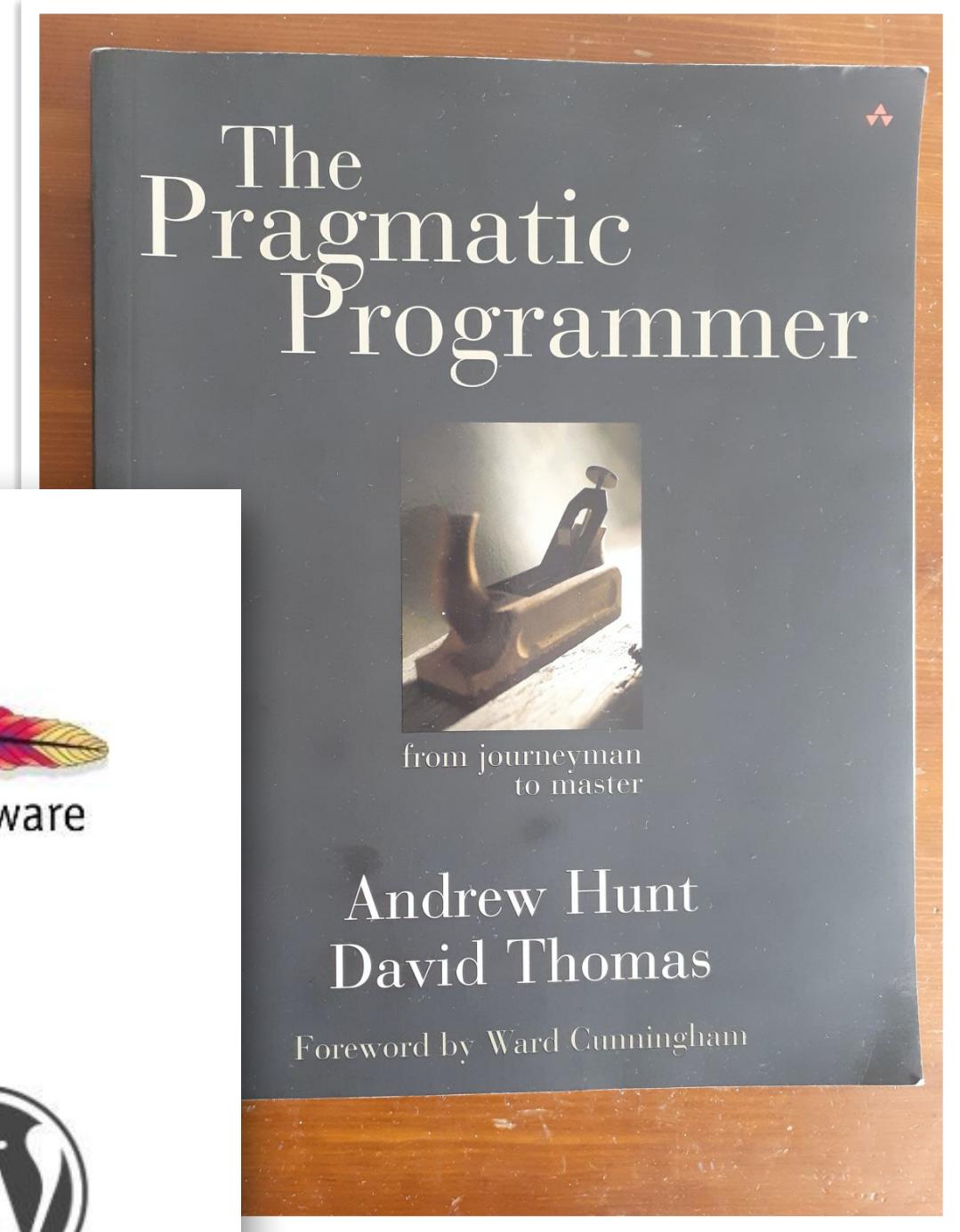
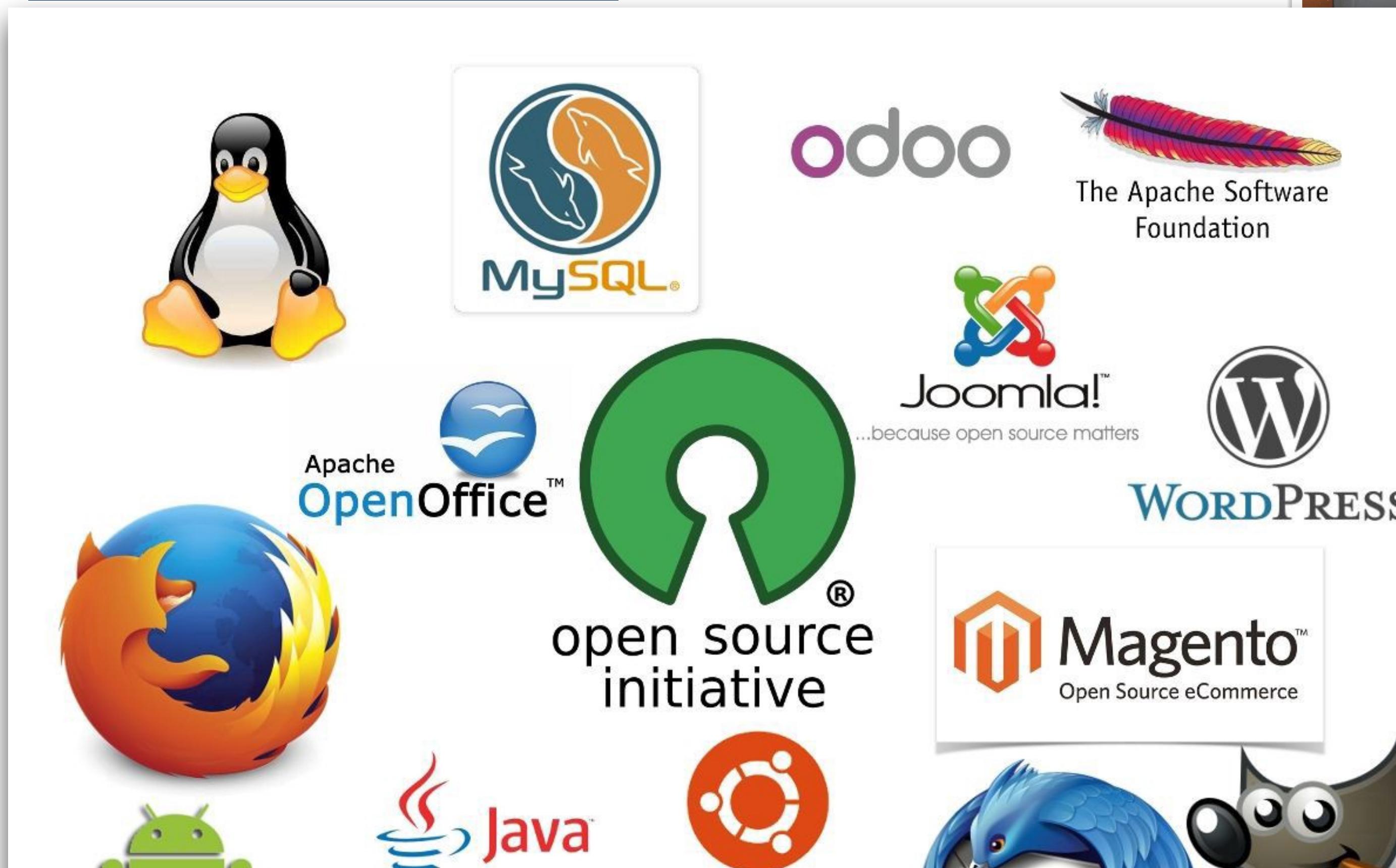
So you've been wandering the web for a while now, and you're ready to start contributing to the great flow of information on the Internet. Where do you start? How do you begin? Providing information on the World Wide Web has two parts: writing the World Wide Web documents and finding a World Wide Web server (such as NCSA Mosaic).



1995



1990s



1990s

Iterative and Incremental Development: A Brief History



Although many view iterative and incremental development as a modern practice, its application dates as far back as the mid-1950s. Prominent software-engineering thought leaders from each succeeding decade supported IID practices, and many large projects used them successfully.

Craig Larman
Valtech

Victor R. Basili
University of Maryland

As agile methods become more popular, some view iterative, evolutionary, and incremental software development—a cornerstone of these methods—as the “modern” replacement of the waterfall model, but its practiced and published roots go back decades. Of course, many software-engineering students are aware of this, yet surprisingly, some commercial and government organizations still are not.

opment” merely for rework, in modern agile methods the term implies not just revisiting work, but also evolutionary advancement—a usage that dates from at least 1968.

PRE-1970

IID grew from the 1930s work of Walter Shewhart,¹ a quality expert at Bell Labs who proposed a series of short “plan-do-study-act” (PDSA)

<https://doi.org/10.1186/s40411-018-0059-z>
Engineering Research and Development (2018) 6:15
<https://doi.org/10.1186/s40411-018-0059-z>

Journal of Software Engineering Research and Development

RESEARCH

Open Access

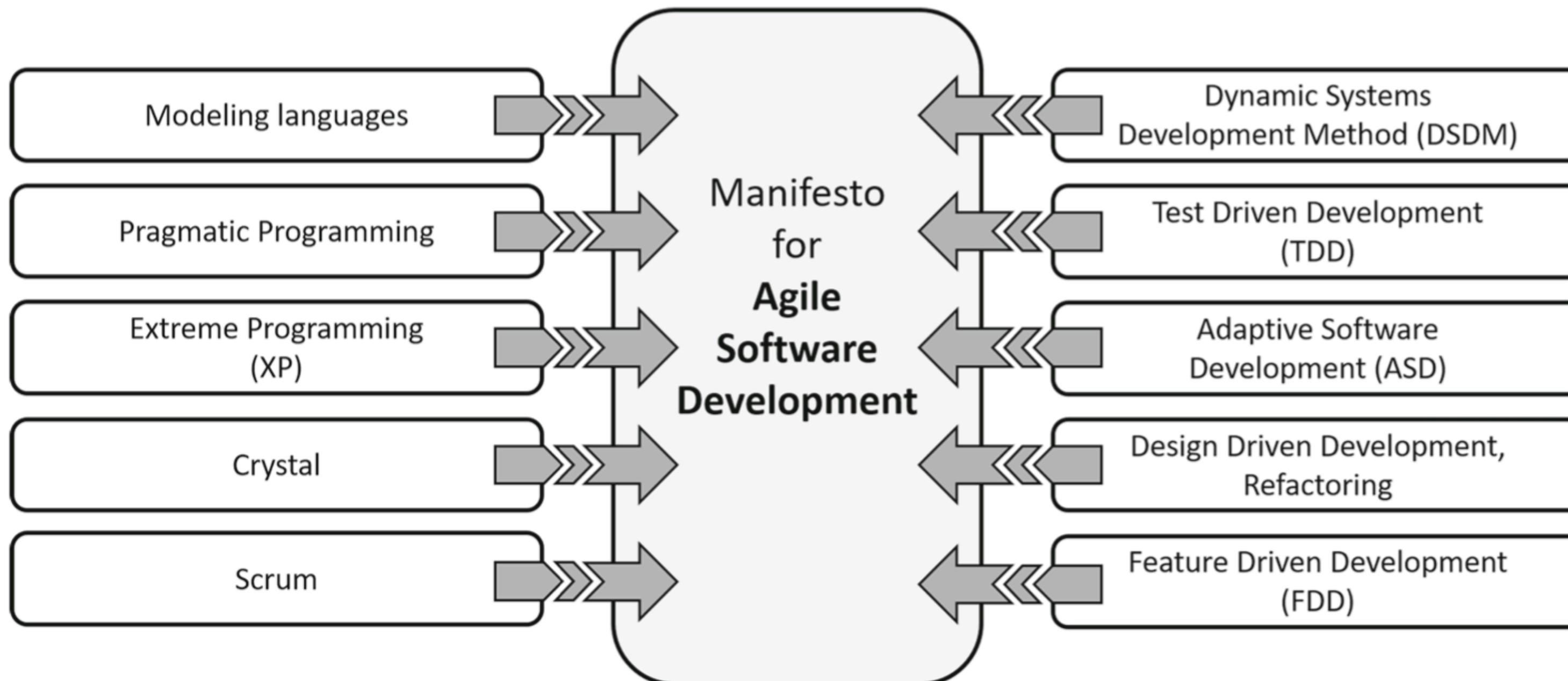


Back to the future: origins and directions of the “Agile Manifesto”—views of the originators

Philipp Hohl^{1*} Jil Klünder², Arie van Bennekum³, Ryan Lockard⁴, James Gifford⁵, Jürgen Münch⁶, Michael Stupperich¹ and Kurt Schneider²

Abstract

In 2001, seventeen professionals set up the manifesto for agile software development. They wanted to define values and basic principles for better software development. On top of being brought into focus, the manifesto has been widely adopted by developers, in software-developing organizations and outside the world of IT. Agile principles and their implementation in practice have paved the way for radical new and innovative ways of software and product development. In parallel, the understanding of the manifesto’s underlying principles evolved over time. This, in turn, may affect current and



2000



Writing The Agile Manifesto

Origins

I can fairly accurately ascribe the origins of the Agile Alliance get together to a retreat held for various leaders in the Extreme Programming community in the Spring of 2000. Kent invited a bunch of active XPers to his rural part of Oregon to discuss various issues in XP. As well as confirmed XPers he also invited a number of people who were interested but separate to XP: such as Alistair Cockburn, Jim Highsmith, and Dave Thomas.

At the meeting we discussed the relationship between XP and other methods of a similar ilk - at the time referred to as Lightweight Methods. We agreed that XP was best as a specific process: "a stake in the ground". We also agreed there was a lot of common ground between XP and some of these other methods. As a result of this (Uncle) Bob Martin decided to try to put together a meeting of people interested in this broader range of methods.

2001

November 20, 2000

Allister Cockbum
Humans and Technology
7691 Dell Road
Salt Lake City, UT 84121
Telephone: 801-947-9275
Fax: 775-416-6457

RE: Light Weight Methods Conference
CONTRACT DUE DATE: November 27, 2000

The following arrangements and conditions will become binding between Snowbird Corporation and Humans and Technology upon signature and receipt to Snowbird by November 27, 2000.

DATES:

Arrival:
Departure:

Sunday, February 11, 2001
Wednesday, February 14, 2001



Snowbird Sales Contact:
Jim Dixon
Executive Meeting Manager
Telephone: 801-933-2272
Fax: 801-933-2298

3. El Manifiesto Ágil

17 Firmantes

Pragmatic
Programmers

- Kent Beck ([Twitter](#))
- Robert C. Martin ([Twitter](#), [Clean Code](#))
- Ward Cunningham ([Twitter](#))
- Ron Jeffries ([Twitter](#), [Blog](#))
- Martin Fowler ([Twitter](#), [Blog](#))
- James Grenning

XP

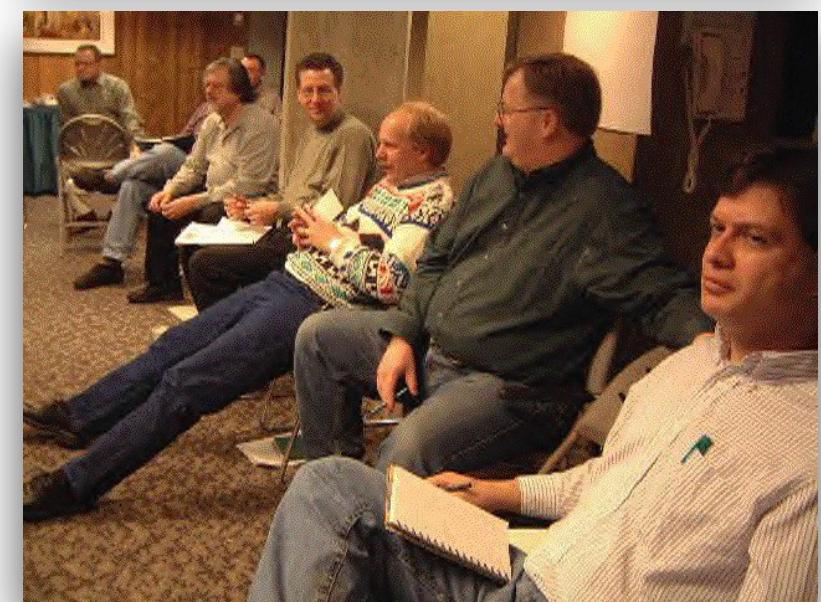
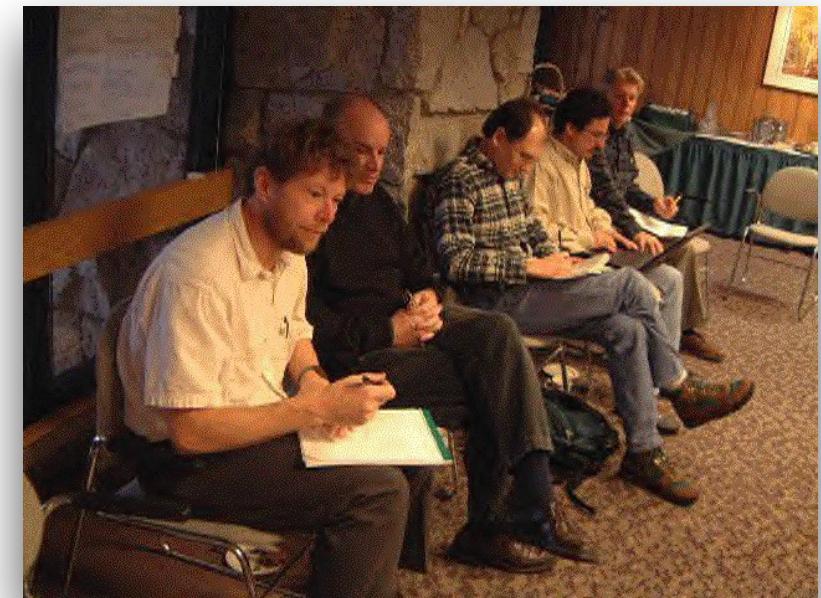
Scrum

- Ken Schwaber ([Twitter](#), [Scrum.org](#))
- Mike Beedle
- Jeff Sutherland ([Twitter](#), [Scrum.org](#))

- Andrew Hunt ([Twitter](#), [Pragmatic Programmer](#))
- Dave Thomas ([Twitter](#), [Pragmatic Programmer](#))
- Alistair Cockburn ([Twitter](#), [Blog](#))

Crystal

- Jon Kern
- Arie van Bennekum
- Jim Highsmith
- Brian Marick
- Steve Mellor



We are uncovering better ways of developing
Software by doing it and helping others do it
Through this work we have come to value

Individuals and interactions
over Process and tools.

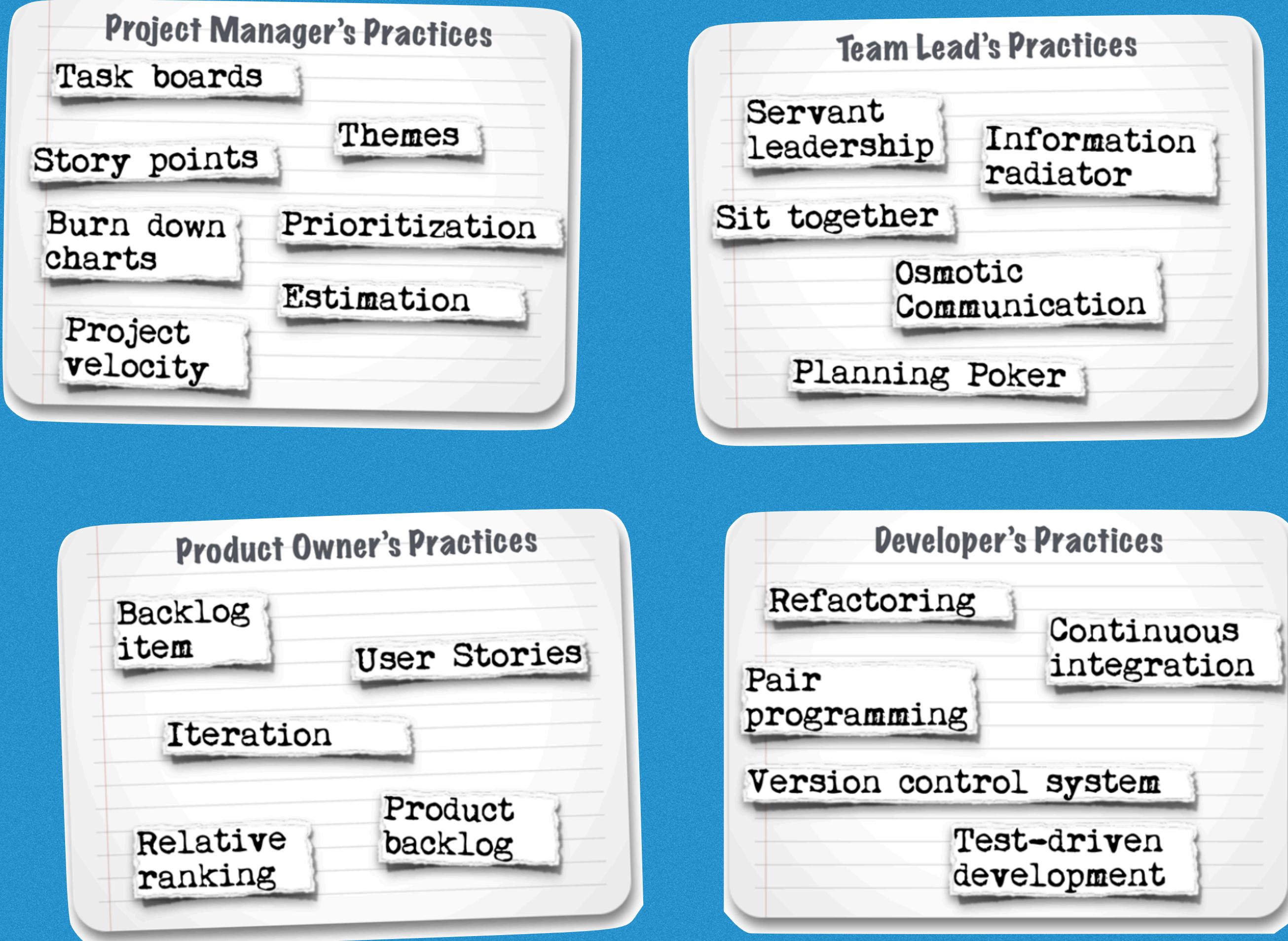
Working Software
over Comprehensive documentation

Customer Collaboration
over Contract negotiation

Responding 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.

Valores



Individuos e interacciones sobre procesos y herramientas

Valores

**Software en funcionamiento
sobre documentación exhaustiva**

Colaboración con el cliente sobre negociación de contratos

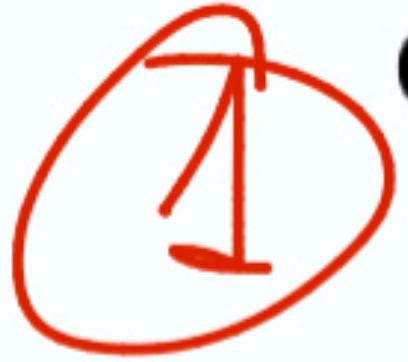
Responder al cambio sobre seguir un plan

Principles behind the Agile Manifesto

We follow these principles:

- ① Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
- ② Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
- ③ Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
- ④ Business people and developers must work together daily throughout the project.
- ⑤ Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.

- ⑥ The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
- ⑦ Working software is the primary measure of progress.
- ⑧ Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
- ⑨ Continuous attention to technical excellence and good design enhances agility.
- ⑩ Simplicity--the art of maximizing the amount of work not done--is essential.
- ⑪ The best architectures, requirements, and designs emerge from self-organizing teams.
- ⑫ At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.



Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.

②

Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.

③

Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.



Business people and developers must work together daily throughout the project.

⑤

Build projects around motivated individuals.
Give them the environment and support they need,
and trust them to get the job done.

6

The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.



Working software is the primary measure of progress.

8

Agile processes promote sustainable development.
The sponsors, developers, and users should be able
to maintain a constant pace indefinitely.



**Continuous attention to technical excellence
and good design enhances agility.**

⑩

Simplicity--the art of maximizing the amount
of work not done--is essential.

11

The best architectures, requirements, and designs
emerge from self-organizing teams.

⑫

At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

Para terminar, dos diapositivas de Henrik Kniberg

- Sacadas de una de sus últimas charlas [Agile intro en KTH](#)

www.agilemanifesto.org
We are uncovering better ways of developing software by doing it and helping others do it.
Through this work we have come to value:

Individuals and interactions over processes and tools

Working solutions over comprehensive documentation

Customer collaboration over contract negotiation

Responding to feedback over following a plan

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

Henrik Kniberg

Find (or create) agile companies!

How to recognize real agility:

- Work in small, cross-functional, self-organizing teams
- Release often & get real user feedback
- Focus on Value rather than Output/Cost
- Experiment a lot with product & process

Beware empty buzzwords

Henrik Kniberg

Para tener en cuenta

- Sacadas de la web



uzzwords



Para terminar, dos diapositivas de Henrik Kniberg

- Sacadas de una de sus últimas charlas [Agile intro en KTH](#)

www.agilemanifesto.org
We are uncovering better ways of developing software by doing it and helping others do it.
Through this work we have come to value:

Individuals and interactions over processes and tools

Working solutions over comprehensive documentation

Customer collaboration over contract negotiation

Responding to feedback over following a plan

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

Henrik Kniberg

Find (or create) agile companies!

How to recognize real agility:

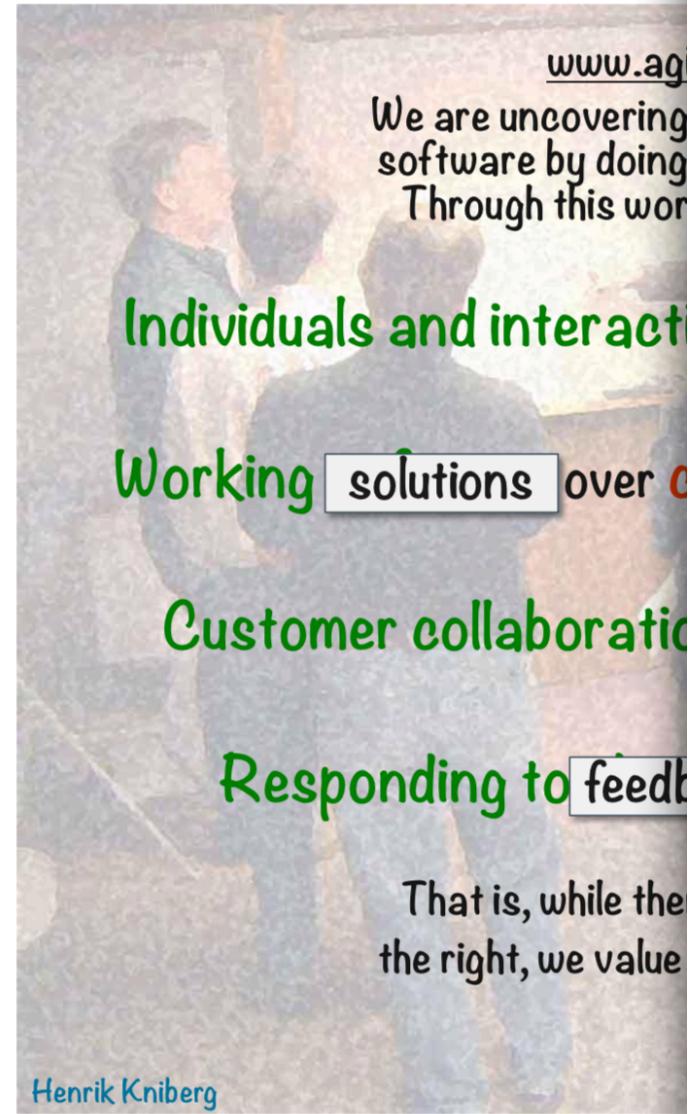
- Work in small, cross-functional, self-organizing teams
- Release often & get real user feedback
- Focus on Value rather than Output/Cost
- Experiment a lot with product & process

Beware empty buzzwords

Henrik Kniberg

Para terminar

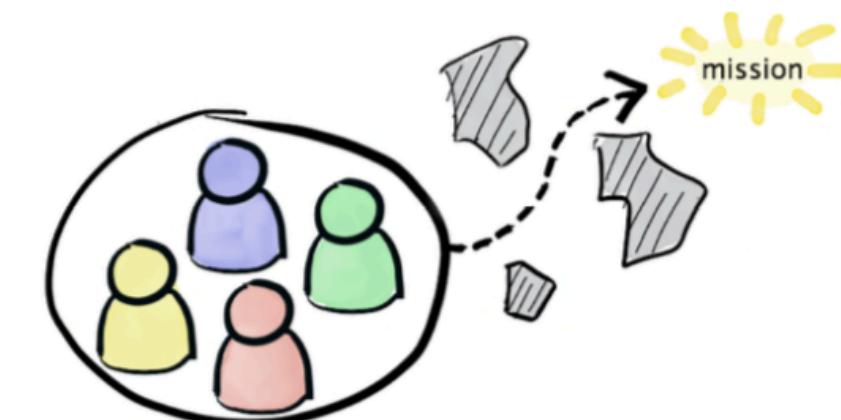
- Sacadas de



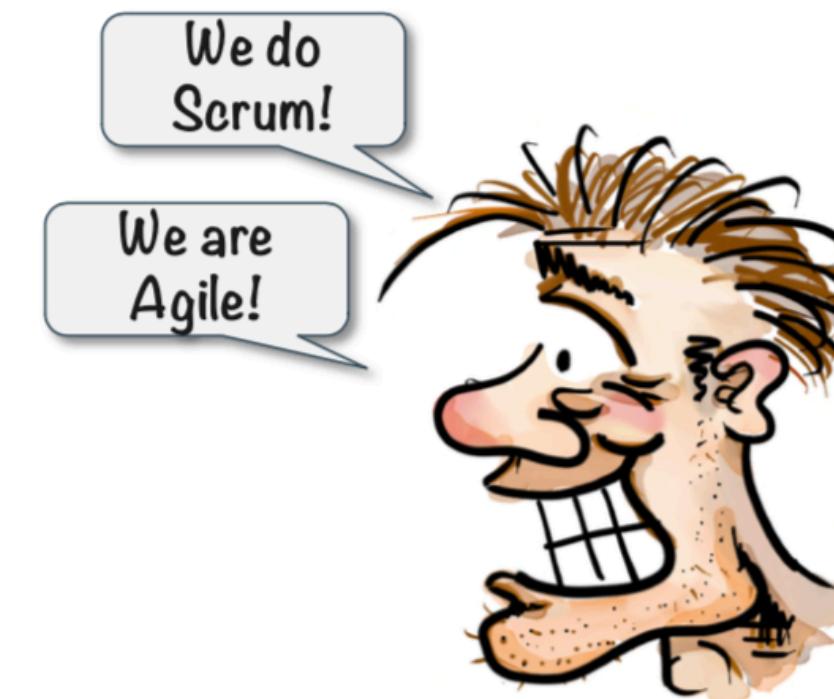
Find (or create) agile companies!

How to recognize real agility:

- Work in small, cross-functional, self-organizing teams
- Release often & get real user feedback
- Focus on Value rather than Output/Cost
- Experiment a lot with product & process



Beware empty buzzwords



Henrik Kniberg

Para terminar, dos diapositivas de Henrik Kniberg

- Sacadas de una de sus últimas charlas [Agile intro en KTH](#)

www.agilemanifesto.org
We are uncovering better ways of developing software by doing it and helping others do it.
Through this work we have come to value:

Individuals and interactions over processes and tools

Working solutions over comprehensive documentation

Customer collaboration over contract negotiation

Responding to feedback over following a plan

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

Henrik Kniberg

Find (or create) agile companies!

How to recognize real agility:

- Work in small, cross-functional, self-organizing teams
- Release often & get real user feedback
- Focus on Value rather than Output/Cost
- Experiment a lot with product & process

Beware empty buzzwords

Henrik Kniberg

master ▾

[apuntes-mads](#) / [apuntes](#) / [manifiesto-agil](#) / [manifiesto-agil.md](#)

domingogallardo Corrección

1 contributor

784 lines (625 sloc) | 36.1 KB

El Manifiesto Ágil

El Manifiesto Ágil, publicado en febrero de 2001, se considera el documento fundacional de todo el nombre.

Establece un conjunto de valores y principios comunes a un número de ideas y corrientes que fueron críticas con el modelo rígido y pesado de desarrollo de software existente en la época.

En esta sesión vamos a poner en contexto el manifiesto, explicando esas ideas previas, para pasar al contenido del mismo.