Executable Modeling: Retrospective and Prospective

Stephen J Mellor

A short history of MDD

UML 2.0: Cast of thousands 2005

Executable UML: Mellor and Balcer 2002

UML 1.1: Three Amigos 1997

Object Lifecycles: Shlaer and Mellor

OMT: Rumbaugh et al 1992

OOA: Shlaer and Mellor 1988

OO Design: Booch 1988

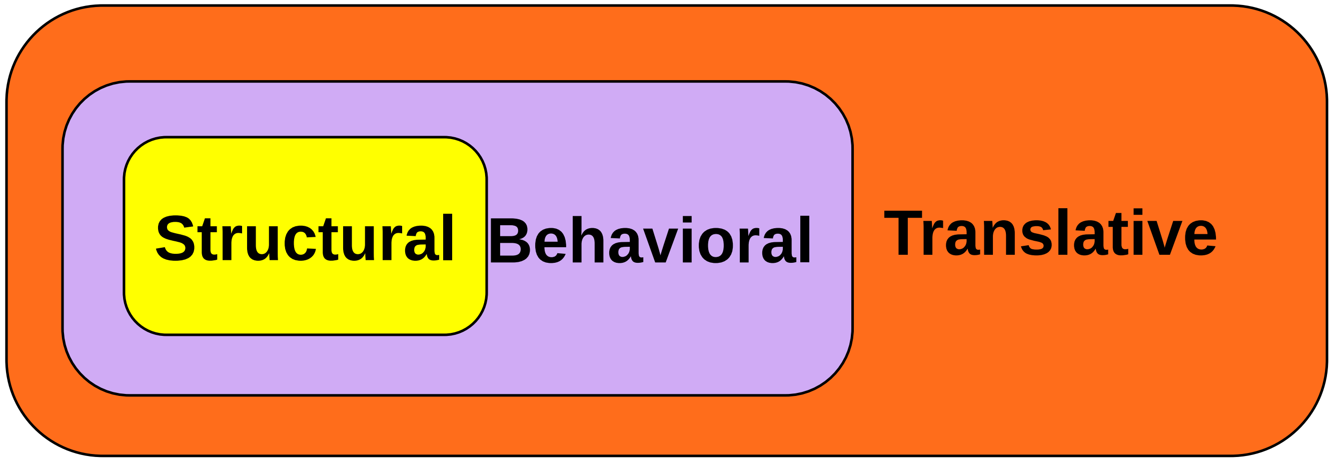
Structured Devpt/RT: Ward and Mellor 1985

Structured Analysis: De Marco 1981

Structured Design: Yourdon and Constantine 1979

Types

In the early Nineties, we received a fax from the OMG requesting participation in a Unified *Method*.



Code Generation from Object Models,
Embedded Systems Programming, March 1998.
Rodney Bell

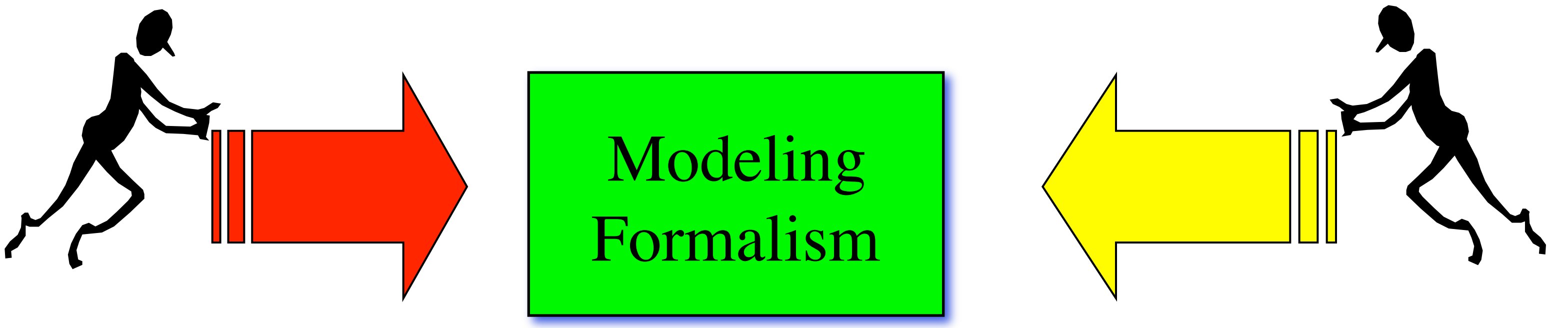
Types

In the early Nineties, w ... eiv ... f ... OMG
requesti

<u>Code Generation from Object Models</u>,
Embedded Systems Programming, March 1998.
Rodney Bell

Two forces

Modeling formalism should be close to the knowledge we' re capturing.



Modeling formalism should be close to the implementation.

Unified Modeling Language

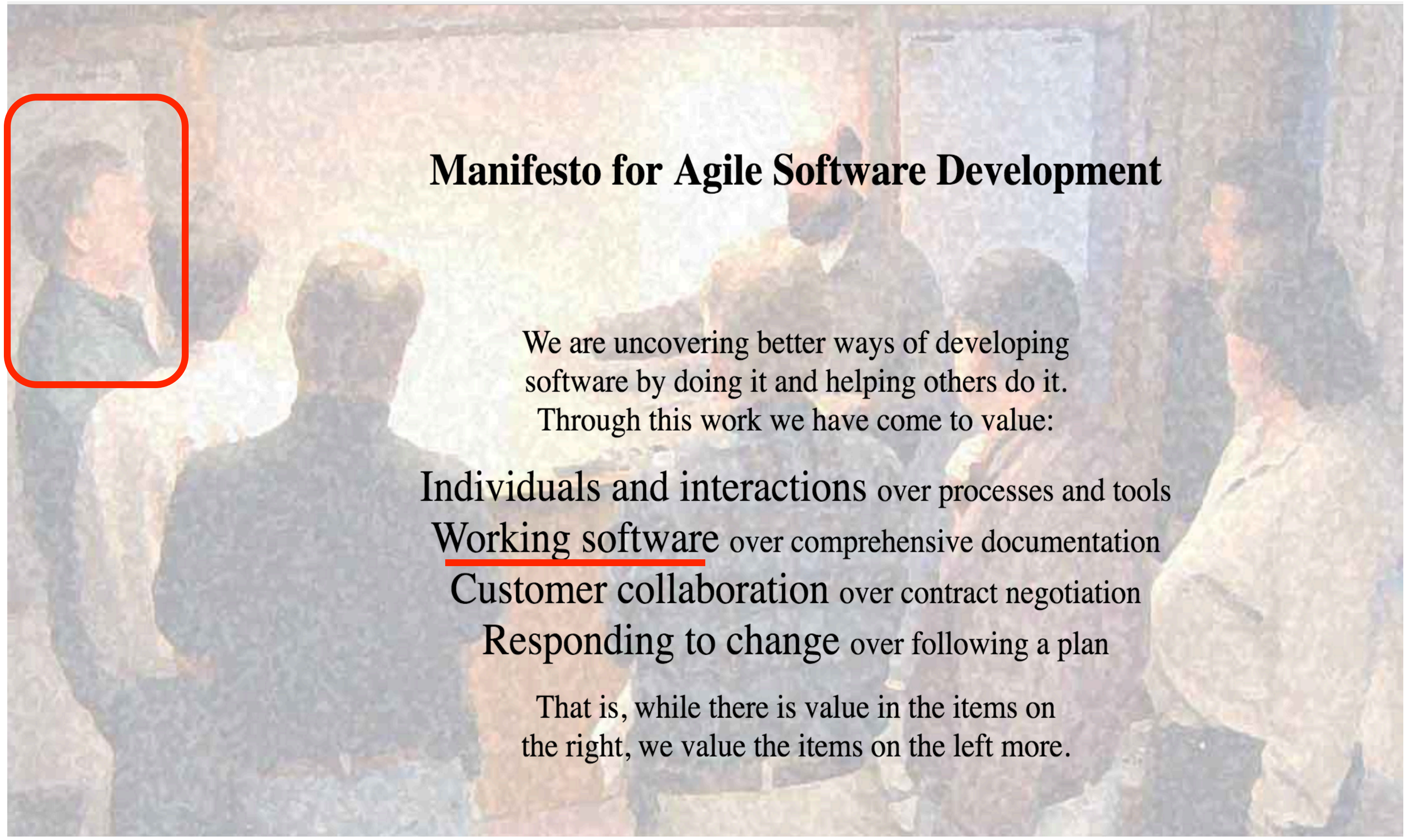
"The Unified Modeling Language is a language for specifying, constructing, visualizing, and documenting the artifacts of a software-intensive system."

The UML Summary

® Object Management Group

UML is ***not*** a method

Agile Manifesto



Manifesto for Agile Software Development

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 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.

A Conversation

The reason code is so important is that it runs, right?

Yes!

An executable model runs, so it can be verified, right?

Yes…

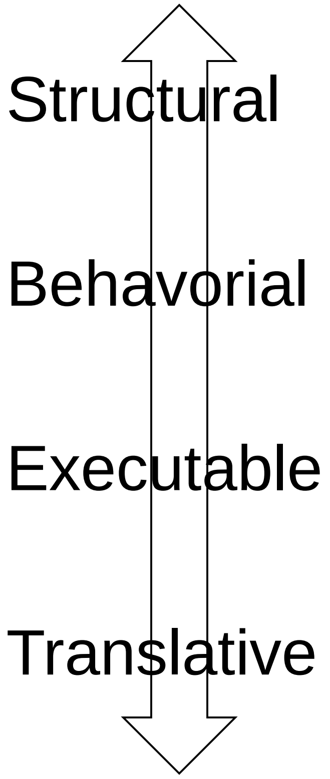
So if a model can be executed, it is as good as code, right?

No. *Code* is the most important thing.

Argh!!!!!!!!

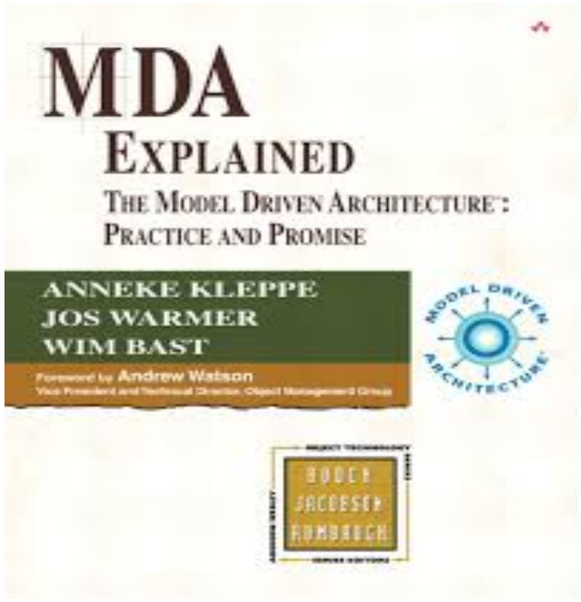
Action Language

- Add code progressively
- Add “model-aware” code progressively
- Add traditional model-aware code
- Add model-based code and translate it



Action Language

Elaborative

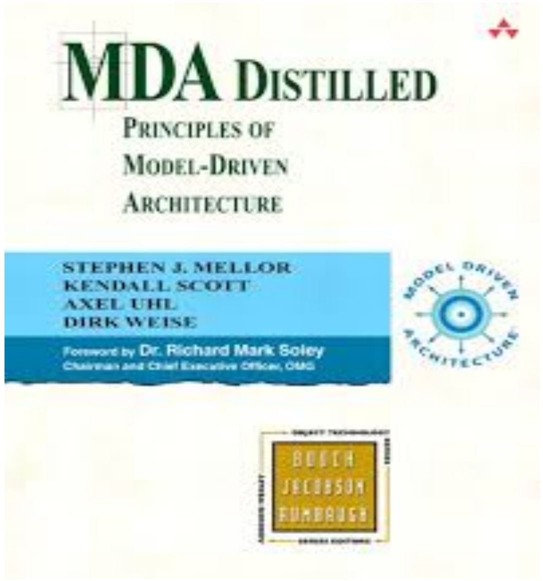

MDA
EXPLAINED
THE MODEL DRIVEN ARCHITECTURE™: PRACTICE AND PROMISE
ANNEKE KLEPPE
JOS WARMER
WIM BAST
Foreword by Andrew Watson
MODEL DRIVEN ARCHITECTURE
BOOCH JACOBSON RUMBAUGH

Structural

Behavorial

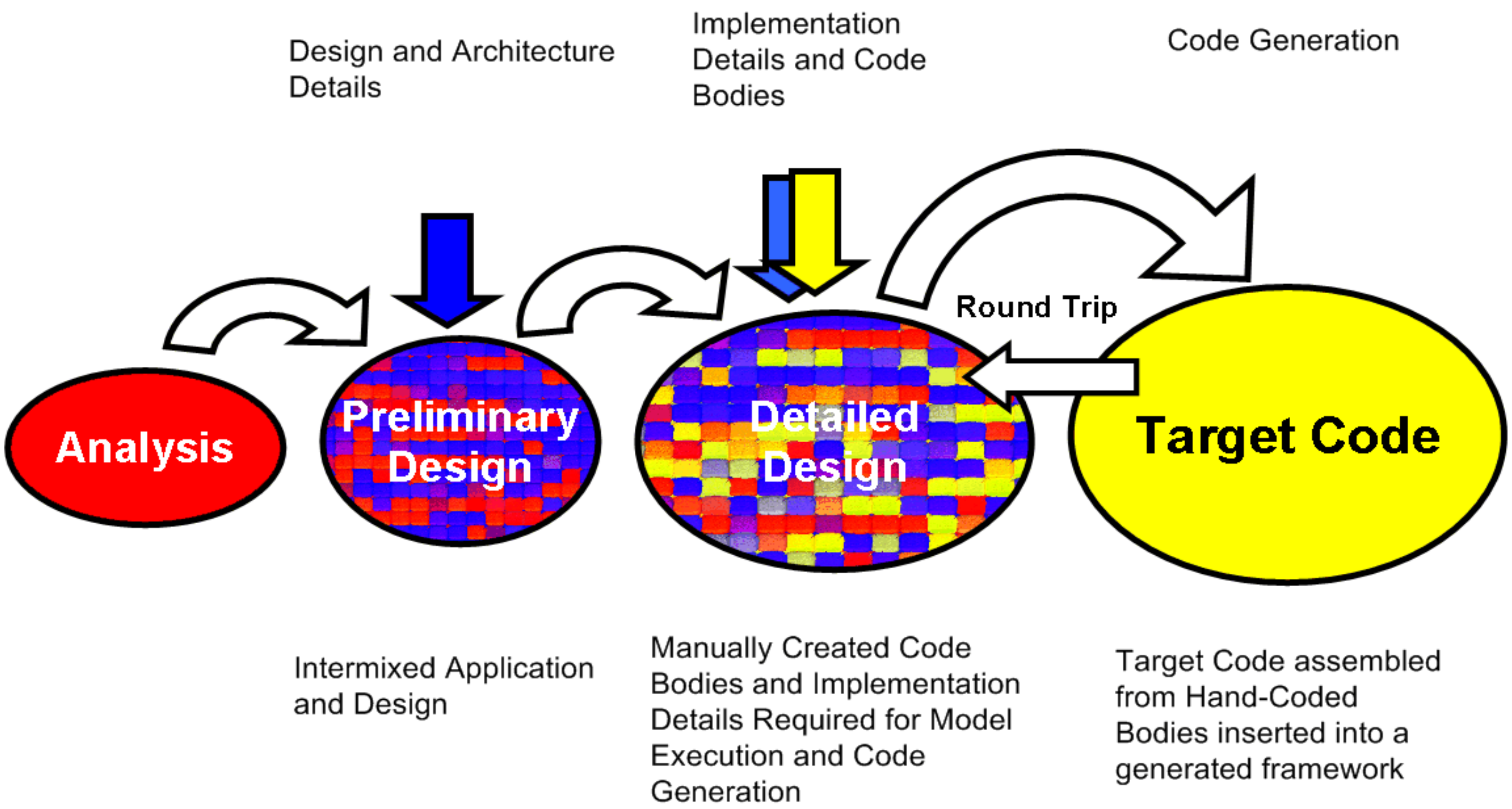
Executable

Translative

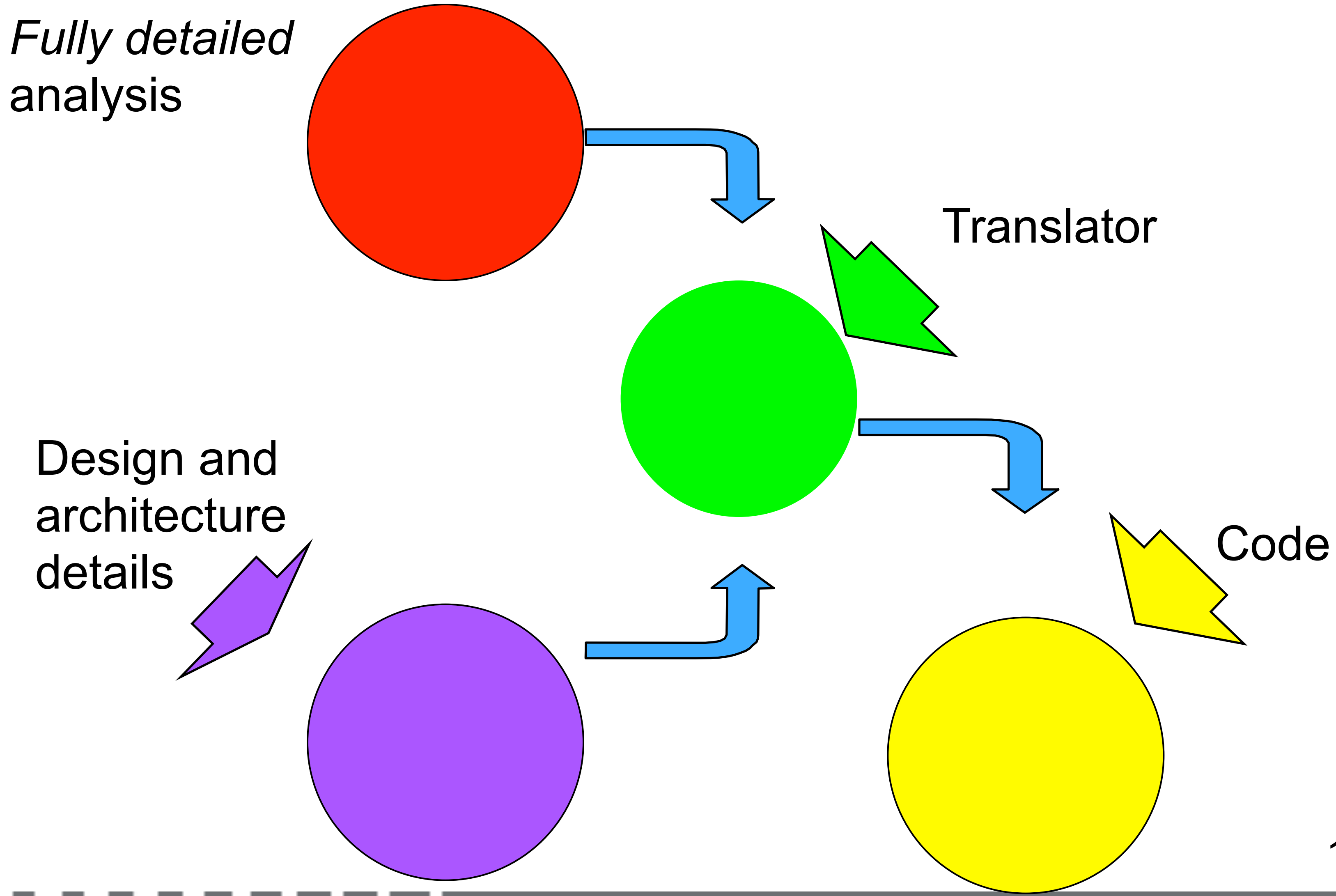

MDA DISTILLED
PRINCIPLES OF MODEL-DRIVEN ARCHITECTURE
STEPHEN J. MELLOR
KENDALL SCOTT
AXEL UHL
DIRK WEISE
Foreword by Dr. Richard Mark Soley
MODEL DRIVEN ARCHITECTURE
BOOCH JACOBSON RUMBAUGH

Executable and Translative

Elaborative development


Design and Architecture Details
Implementation Details and Code Bodies
Code Generation
Round Trip
Analysis
Preliminary Design
Detailed Design
Target Code
Intermixed Application and Design
Manually Created Code Bodies and Implementation Details Required for Model Execution and Code Generation
Target Code assembled from Hand-Coded Bodies inserted into a generated framework

Change the assumptions….

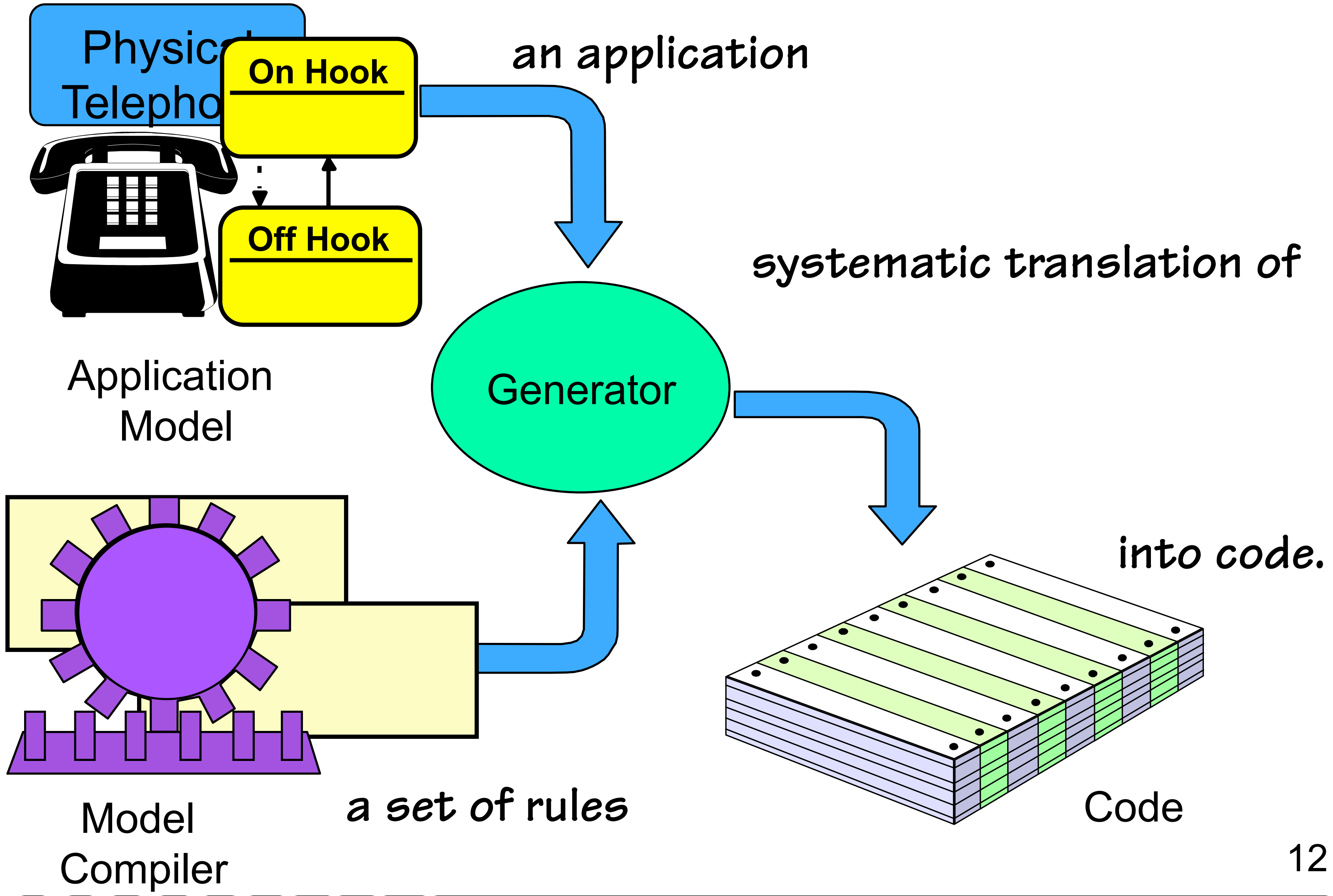


“Recursive Design”

Recursive Design views system design as

- ☞ **a process of *systematic translation* of**
  - ☞ **an application**
    - ☞ **according to a set of rules**
      - ☞ **into code.**

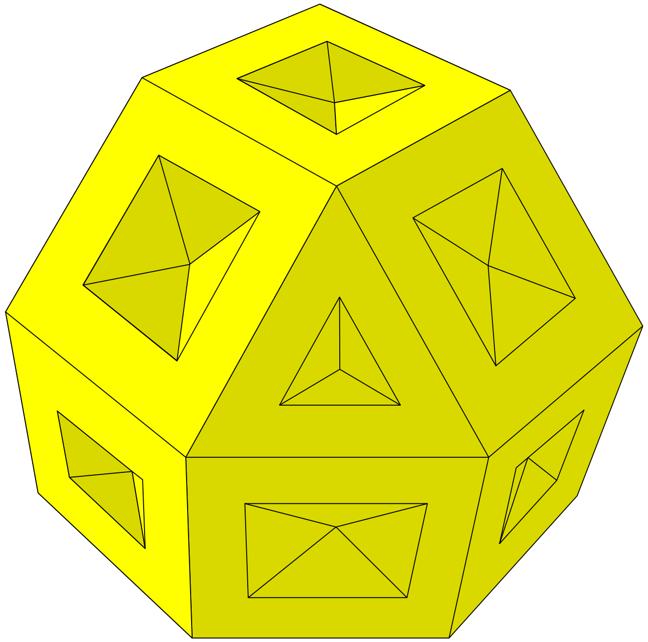
Recursive Design...



Uniformity

A minimal, uniform set of organization rules:

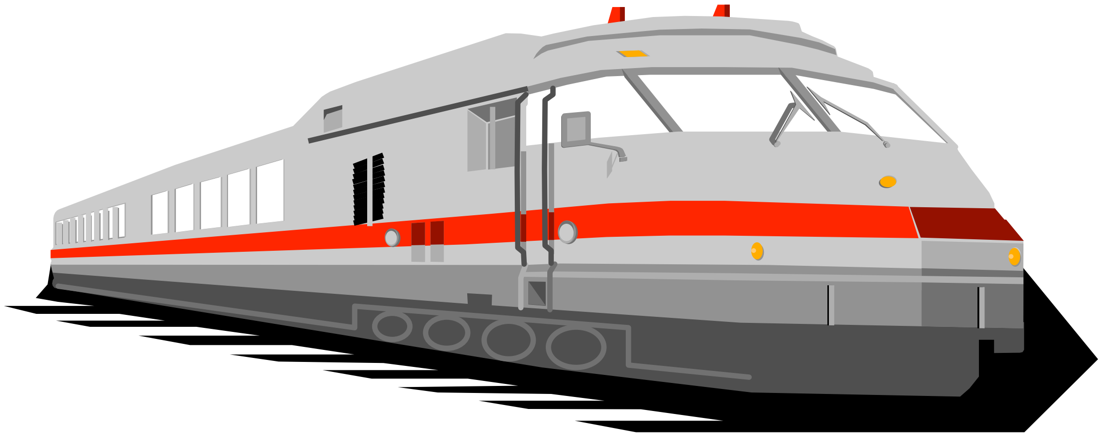
- reduces cost of understanding, building, and maintaining the software
- decreases integration effort
- leads to smaller, more robust code

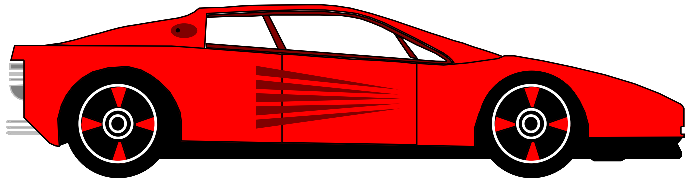


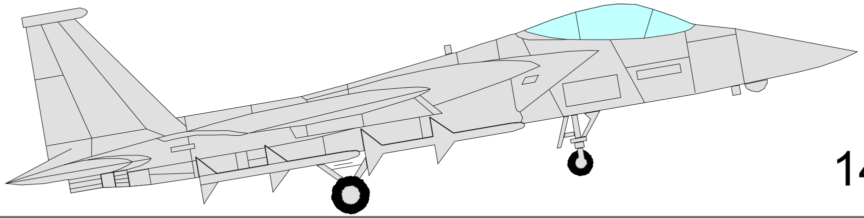
This uniform set of organization rules is a *software architecture*.

Multiple Architectures

There may be several implementations with different performance properties.

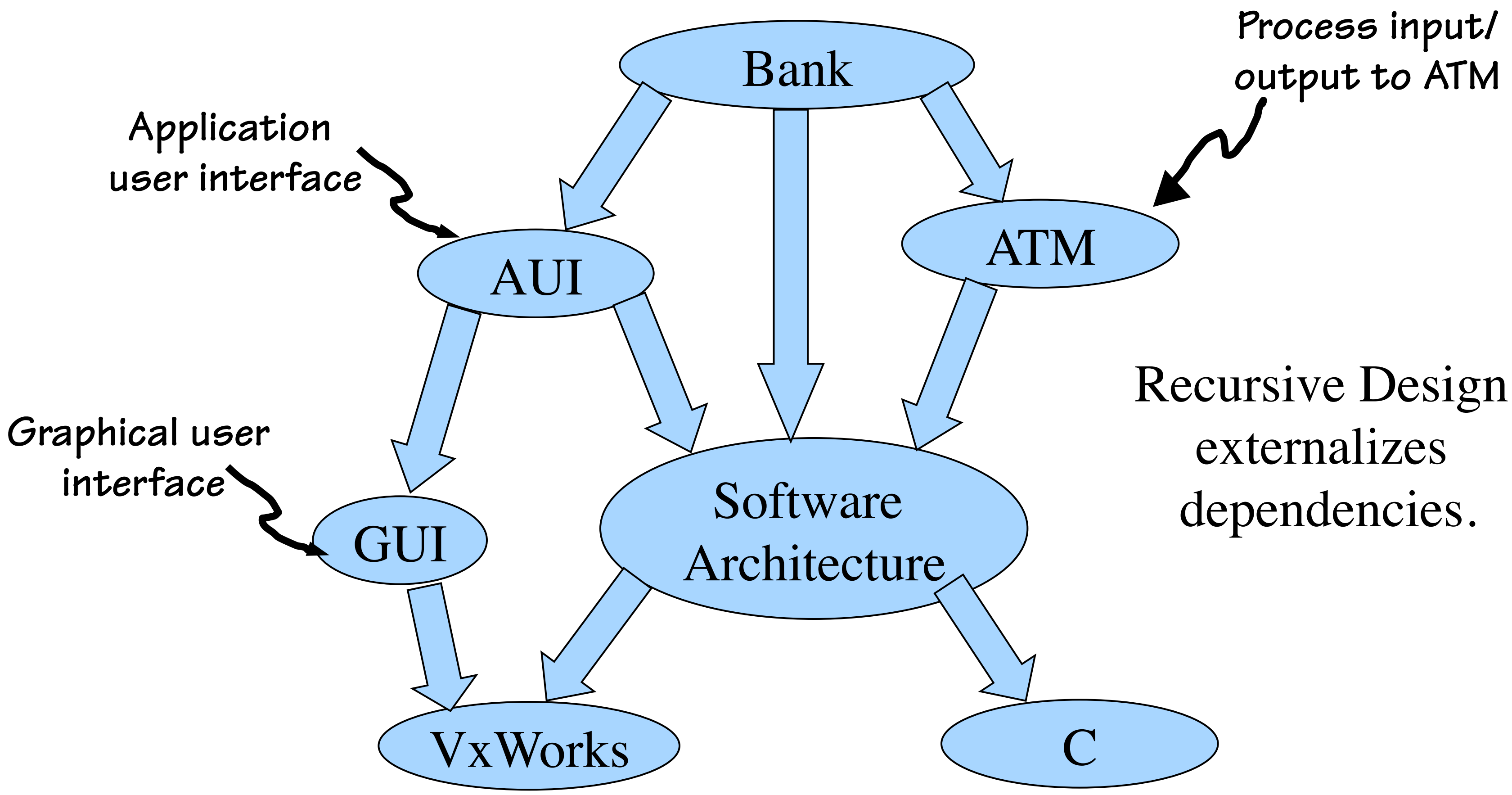






Separation of Subject Matter

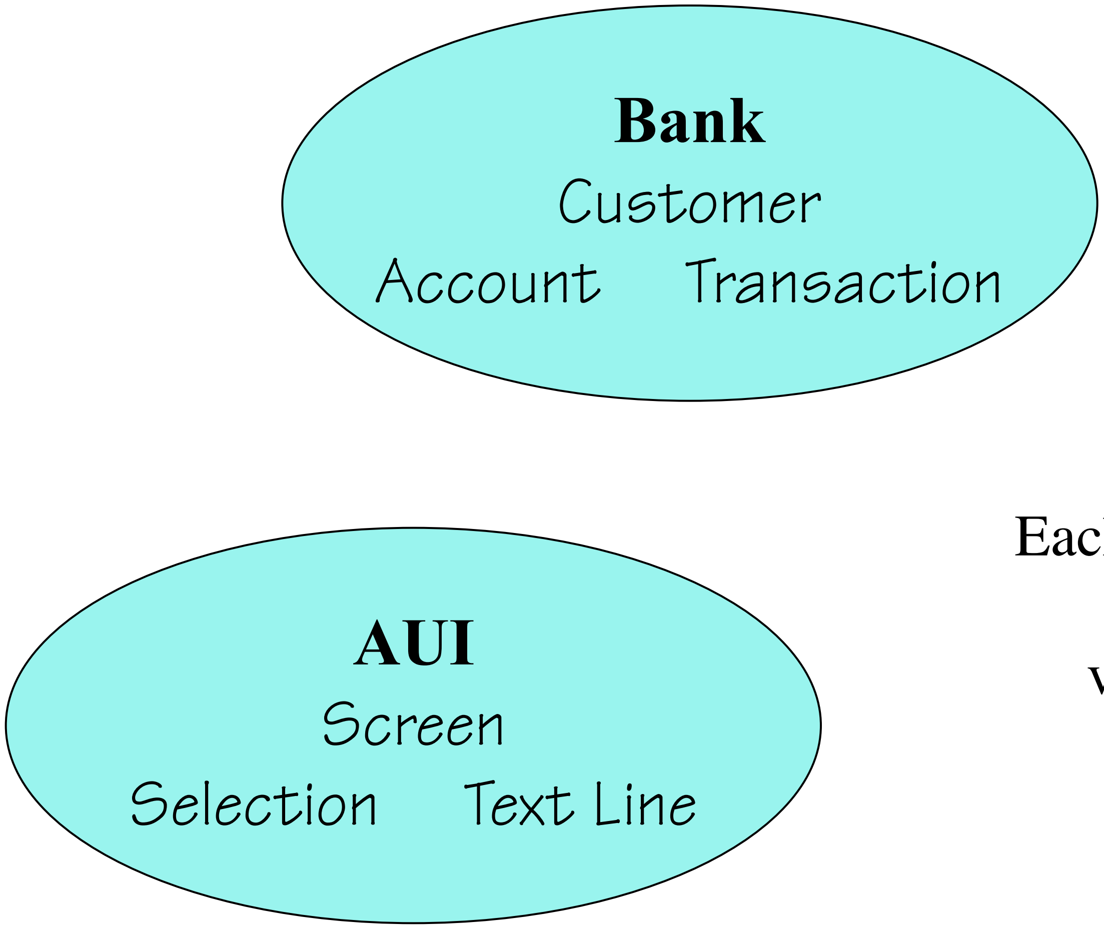
A system comprises several different subject matters.



Recursive Design externalizes dependencies.

Domains

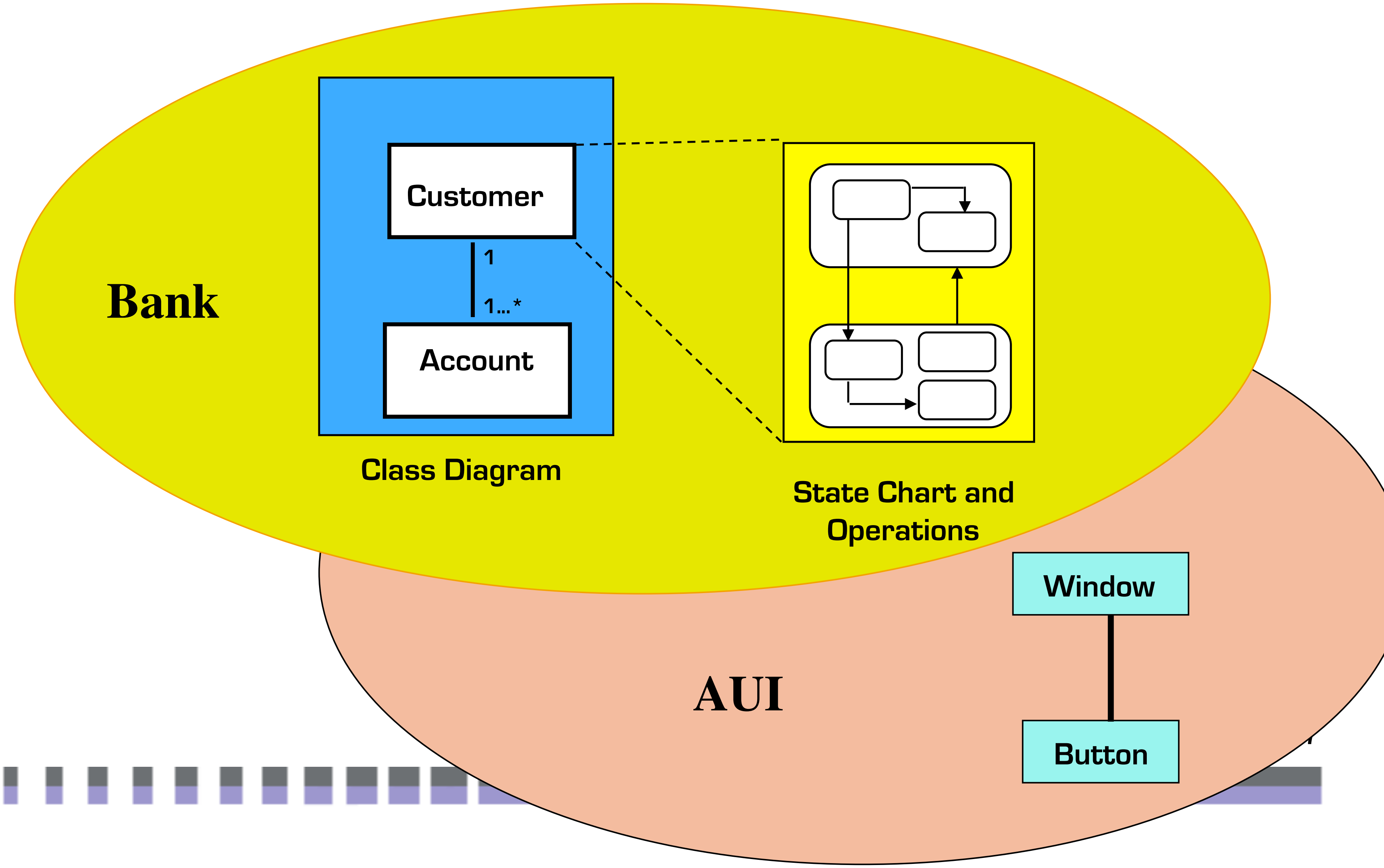
Each subject matter is a problem domain.



Each domain has its own vocabulary.

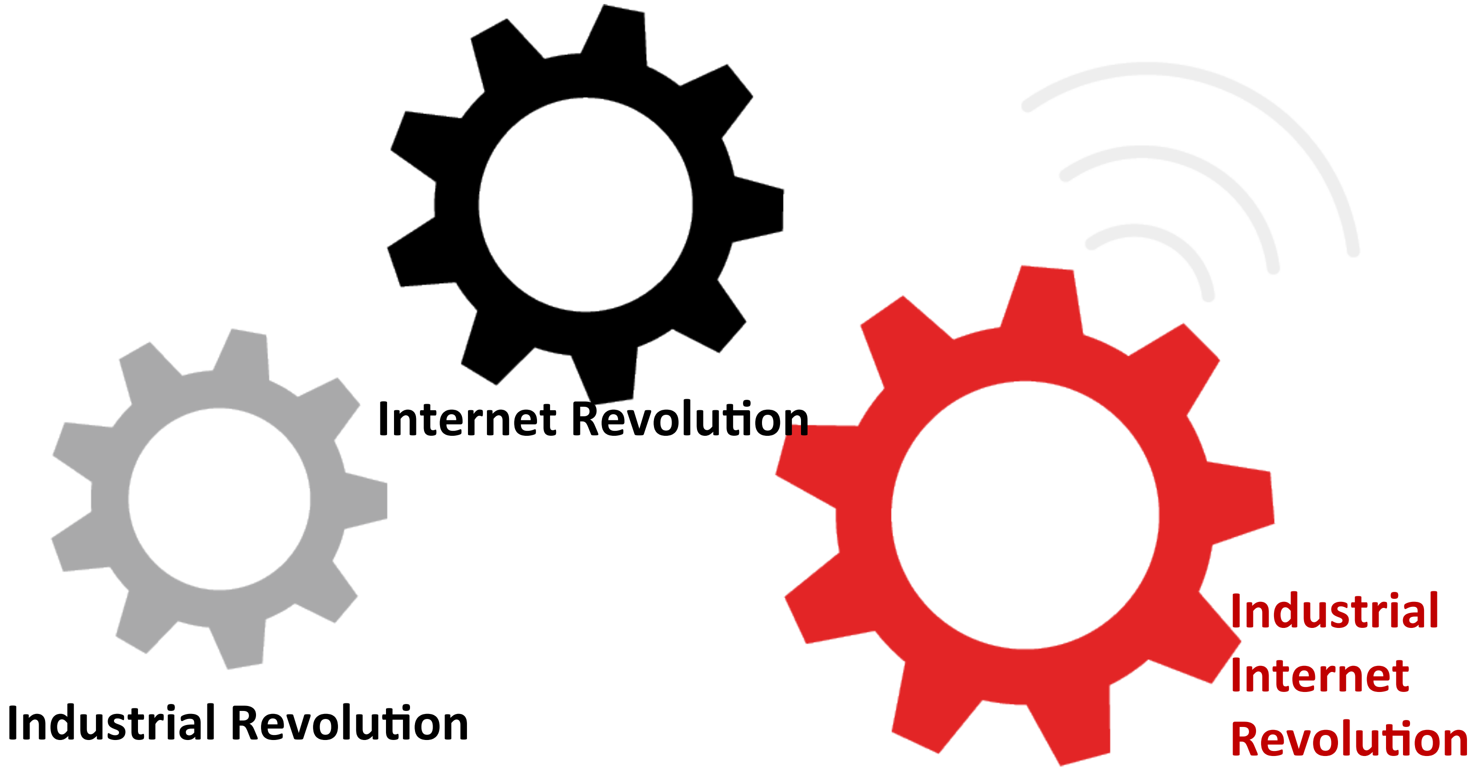
Composable Domains

Model each domain, then connect them.



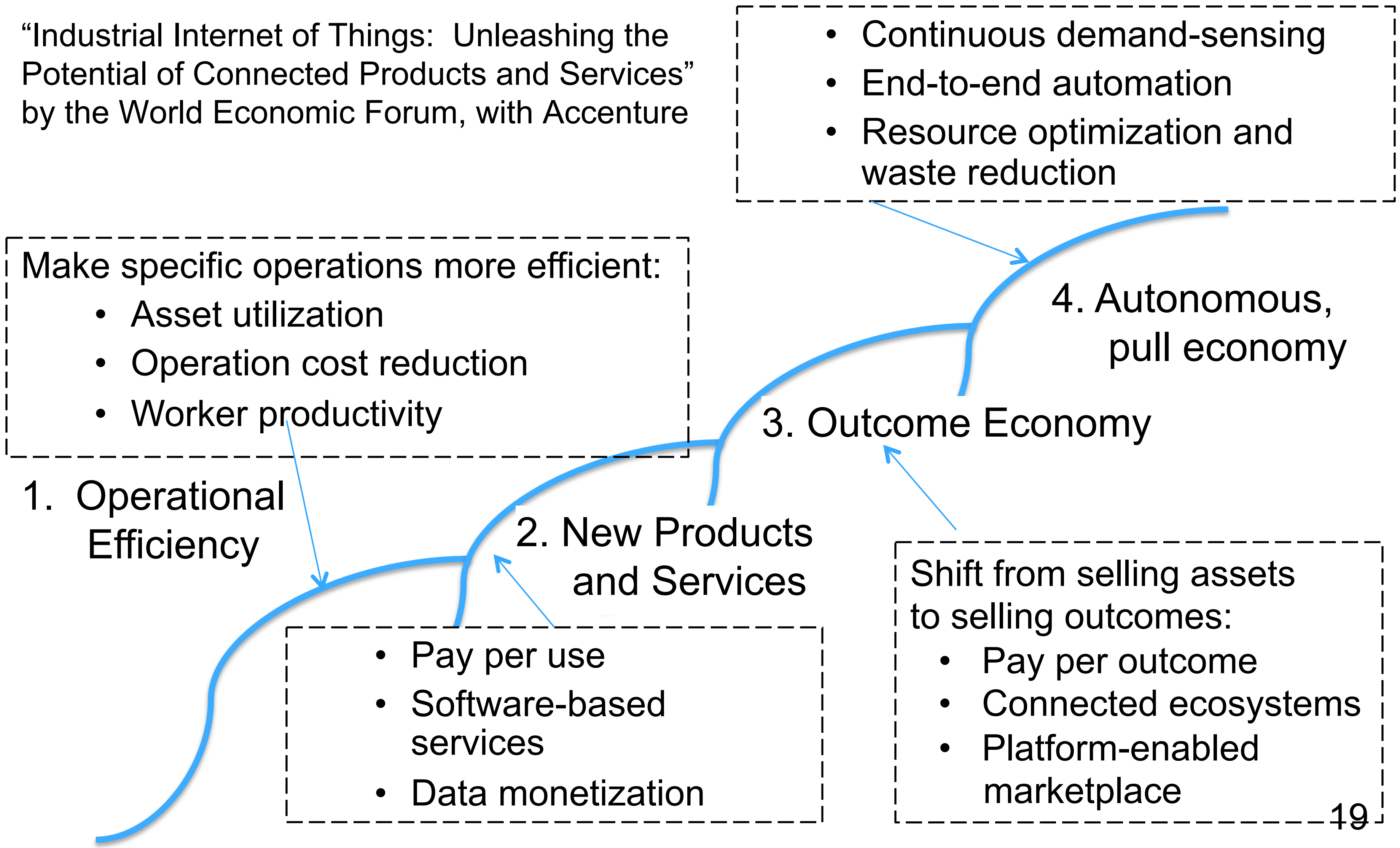
The Industrial Internet

It's an internet of things, machines, computers and people, enabling intelligent industrial operations using advanced data analytics for transformational business outcomes.



The Future

"Industrial Internet of Things: Unleashing the Potential of Connected Products and Services" by the World Economic Forum, with Accenture



1. Operational Efficiency
2. New Products and Services
3. Outcome Economy
4. Autonomous, pull economy

Make specific operations more efficient:

- Asset utilization
- Operation cost reduction
- Worker productivity

- Pay per use
- Software-based services
- Data monetization

Shift from selling assets to selling outcomes:

- Pay per outcome
- Connected ecosystems
- Platform-enabled marketplace

- Continuous demand-sensing
- End-to-end automation
- Resource optimization and waste reduction

When will be Executable Models be Commonplace?

1985: “In three years time…”
1987: “In three years time…”
1989: “In three years time…”
1991: “In three years time…”
1993: “In three years time…”
1995: “In three years time…”
1997: “In three years time…”
1999: “In three years time…”
2001: “In three years time…”
2003: “In three years time…”

Thank you

StephenMellor@StephenMellor.com