COUPLING AND COHESION

PV260 – SOFTWARE ARCHITECTURE PRIMER

Ondra @ Y Soft
April 19th, 2022

YSOFT APPLIED™ RESEARCH

WHAT TO REMEMBER?

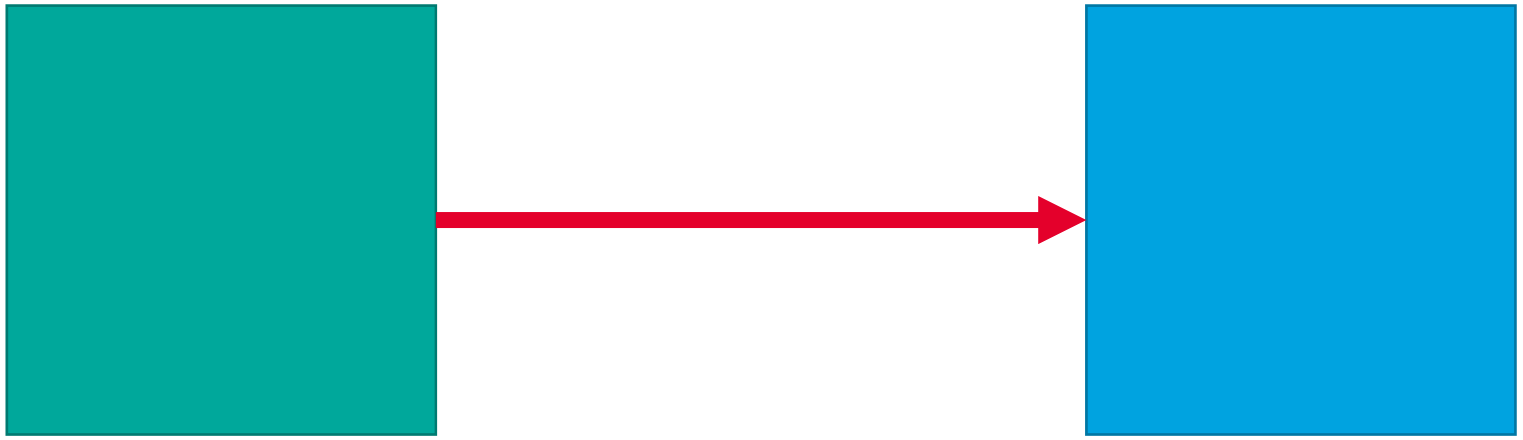
DESIGN FOR FLOW OF DATA

DISCLAIMER: COUPLING AND COHESION ARE…

MEASURABLE PROPERTIES, NOT PRINCIPLES

COUPLING

- A “DEGREE” OF DEPENDENCE OR INTER-DEPENDENCE



A MODULE/SERVICE/COMPONENT/UNIT

- A PART OF A SYSTEM, A SYSTEM IN ITS OWN RIGHT
- COMPILE TIME LEXICAL UNITS, STATICALLY LINKED LIBRARIES, DYNAMICALLY LINKED LIBRARIES, (MICRO-)SERVICES, 1ST CLASS FUNCTIONS, ETC.

A CONNECTION

- A SPECIFIC DIRECTED DEPENDENCY BETWEEN MODULES (SYMBOL EXPORTED FROM A LEXICAL SCOPE, ENTRY POINT CALL, SERVICE CALL)
- A REFERENCE TO AN ELEMENT RESIDING IN ANOTHER MODULE (A POINTER TO A SHARED MEMORY, A REFERENCE TO A CLIENT STUB OF A REMOTE TYPE, A REFERENCE TO A REMOTE SERVICE, ETC.)
- INTER-CONNECTION IS A BI-DIRECTIONAL DEPENDENCY ($A \rightarrow B \wedge B \rightarrow A$)
- MULTI-CONNECTION IS MULTIPLE DEPENDENCY BETWEEN TWO MODULES ($A \rightarrow B \wedge A \rightarrow B$)

IMPACT OF COUPLING?

- CORRELATES WITH PROBABILITY OF PROPAGATION
- PROPAGATION OF CHANGES OR (PARTIAL) FAILURES BOTH IN THE SOURCE CODE AND AT RUNTIME
- COUPLING ~ COST OF SCALING AND COST OF CHANGE

A KEY QUESTION?

- HOW MUCH OF ONE MODULE INTERNALS NEEDS TO BE KNOWN IN ORDER TO UNDERSTAND A CONNECTED MODULE(S)

IMPACT OF COUPLING?

FACTORS THAT INFLUENCE COUPLING OF MODULES

Connection Type

- Minimally Connected Modules (Ideal)
- Normally Connected Modules (Pragmatic)
- Pathologically Connected Modules (Problematic, not minimal, not normal)

Structural Complexity

- Complexity of the contract structure
- Structure of data elements (not data volume) which are part of the data flow
- Number of callable functions and their arguments (entry points)

Type of Information Flow

- Data (Input-Output)
- Control
- Hybrid

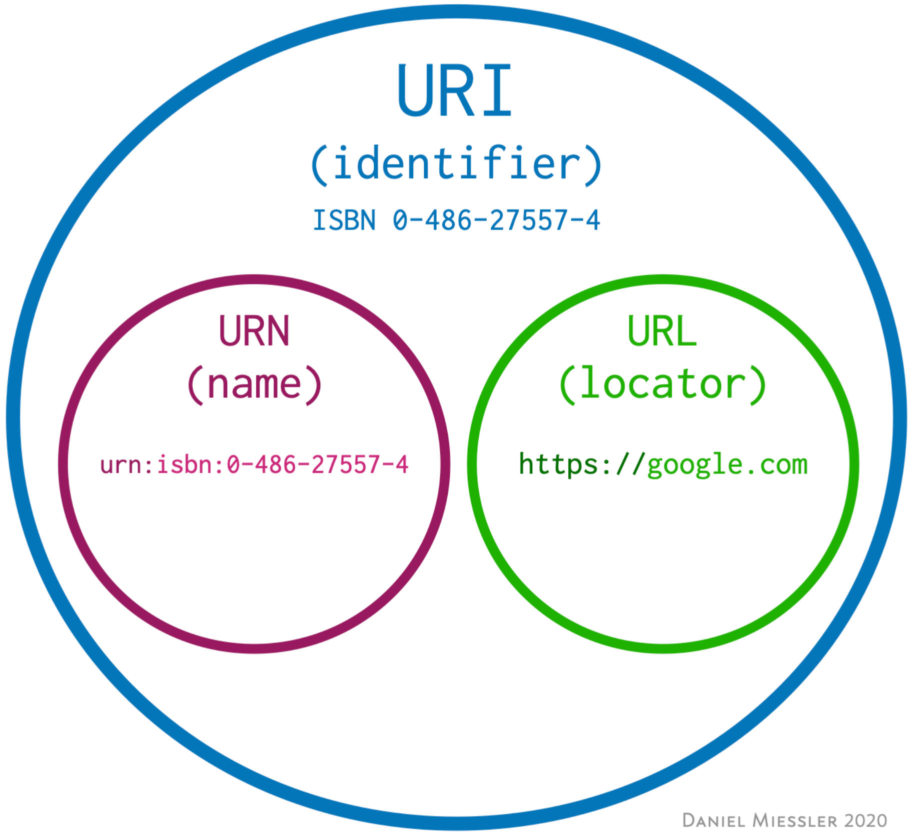
Binding

- Early
- Late
- *What* are we binding in the world of micro-services?

MINIMALLY CONNECTED MODULES

ONE CONNECTION ENDPOINT / ENTRY POINT PER MODULE

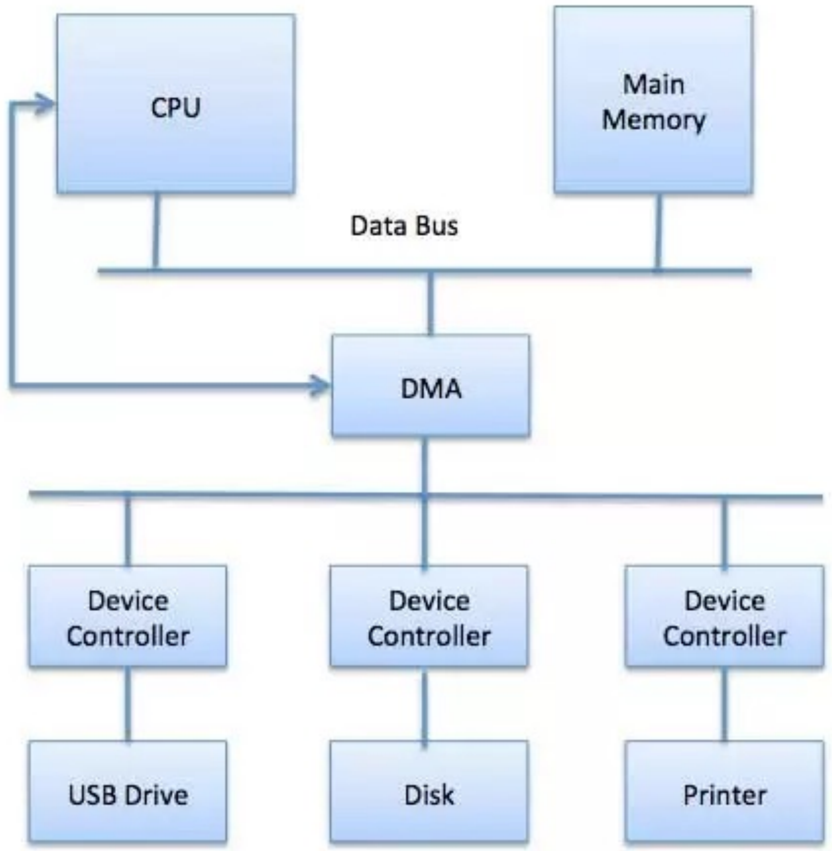
- Data flow into a module
- Data flow out of a module
- Control flow reception by a module
- Control flow transition from a module
- Identity (example of identity without data/control flow?)



CONTROL VS. DATA FLOW

EXAMPLE: DIRECT MEMORY ACCESS (DMA)

- Control Flow: CPU → DMA Controller, DMA → Device Controller
- Data Flow: Device Controller → DMA Controller → Main Memory



NORMALLY CONNECTED MODULE

- **MINIMALLY CONNECTED MODULE, WHICH…**
  - Can have multiple inbound connections as long as they are minimal data-flow connections (data flows transfer complete state).
  - If there is a transfer and return of control, the control returns only to an endpoint explicitly defined by the calling module.
  - There is a transfer of control which does not provide "return" endpoint (uni-directional message, request without expecting reply, etc.).

NORMAL AND MINIMALLY CONNECTED SYSTEMS

…ARE FOR ALL PRACTICAL INTENTS AND PURPOSES EQUAL

FLOW TYPE & COUPLING

FROM WORST TO BEST

- **Content Coupling** means, that the modules are content-interleaved, i.e. one exists as part of the other (lifecycle, codebase), one shares a (part of) state-bearing infrastructure with the other (such as database), etc.
- **Hybrid Coupling** occurs when one module's data are another module's control.
- **Control Coupling** occurs when one module passes data to another module and the receiving module makes control flow decision (dispatch).
- **Stamp Coupling** is a form of data coupling, when a module receives data it does not need (i.e. a superset of data it is processing).
- **Data (Input/Output) Coupling** occurs when data (without any control flow/dispatch flags are transferred).

REASONING ABOUT COUPLING

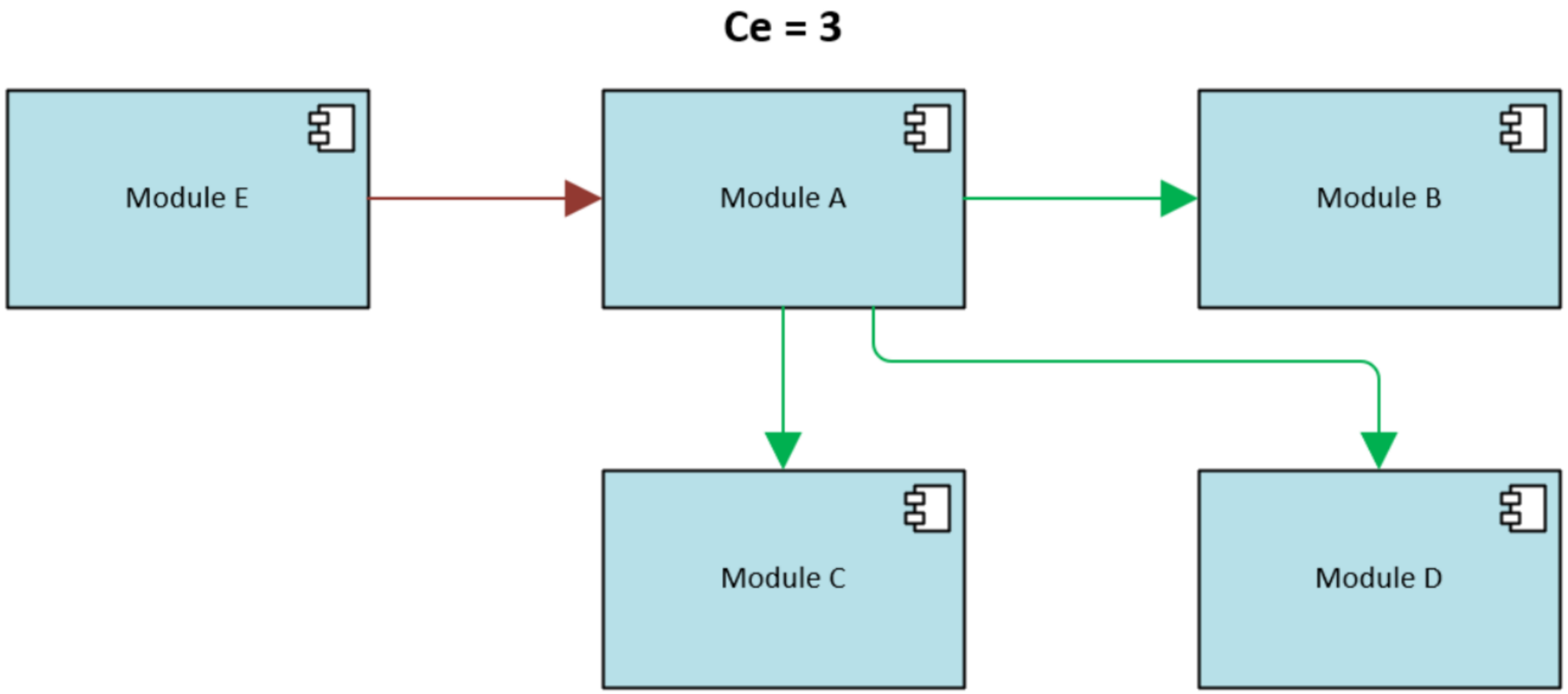
COST OF SCALING

EFFERENT, AFFERENT COUPLING AND INSTABILITY

- CE (EFFERENT COUPLING) = #OF OUTGOING CONNECTIONS
- CA (AFFERENT COUPLING) = #OF INCOMING CONNECTIONS

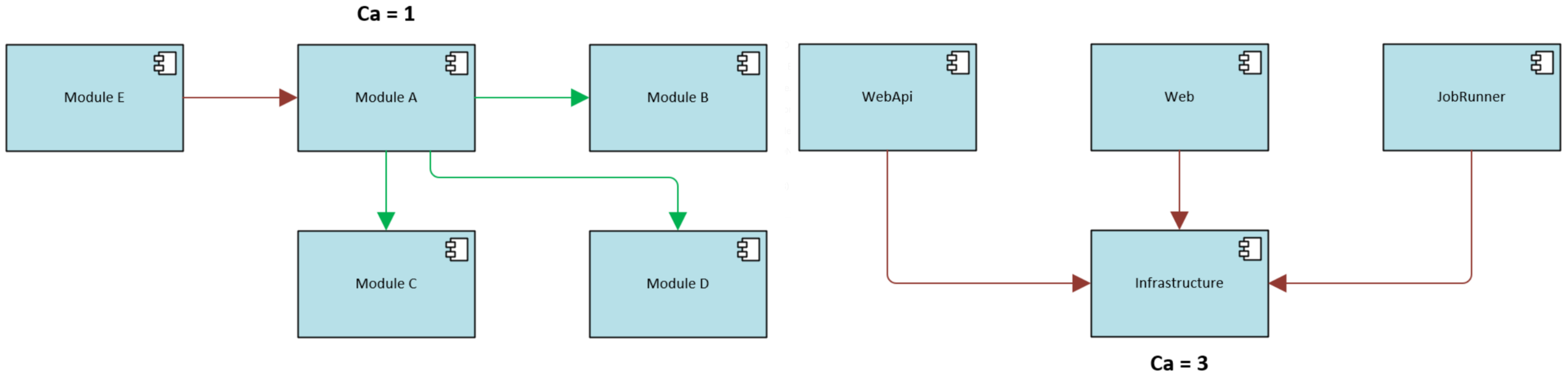
EFFERENT COUPLING

- CE (EFFERENT COUPLING) = #OF OUTGOING CONNECTIONS



AFFERENT COUPLING

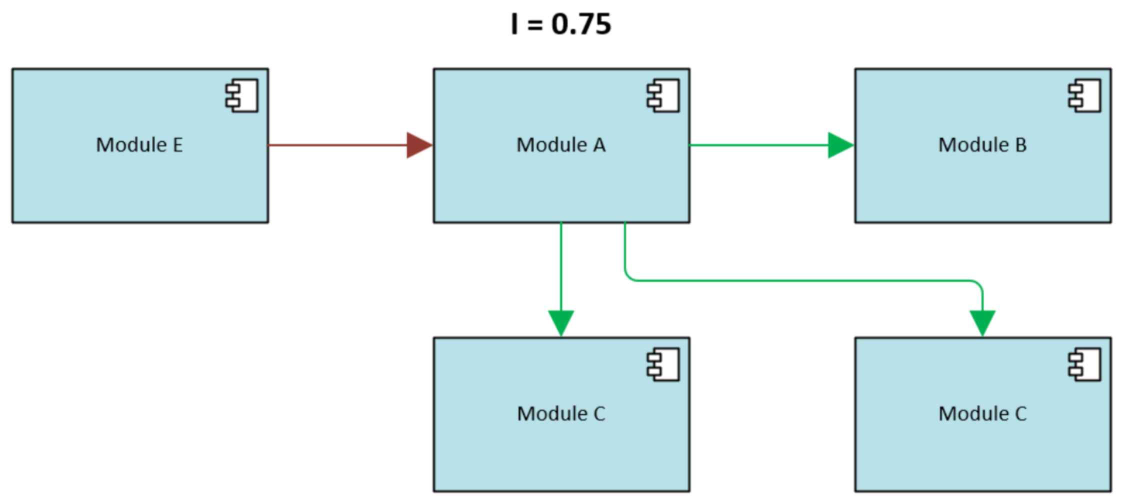
- CA (AFFERENT COUPLING) = #OF INCOMING CONNECTIONS



INSTABILITY

▪LOWER IS BETTER ;-)

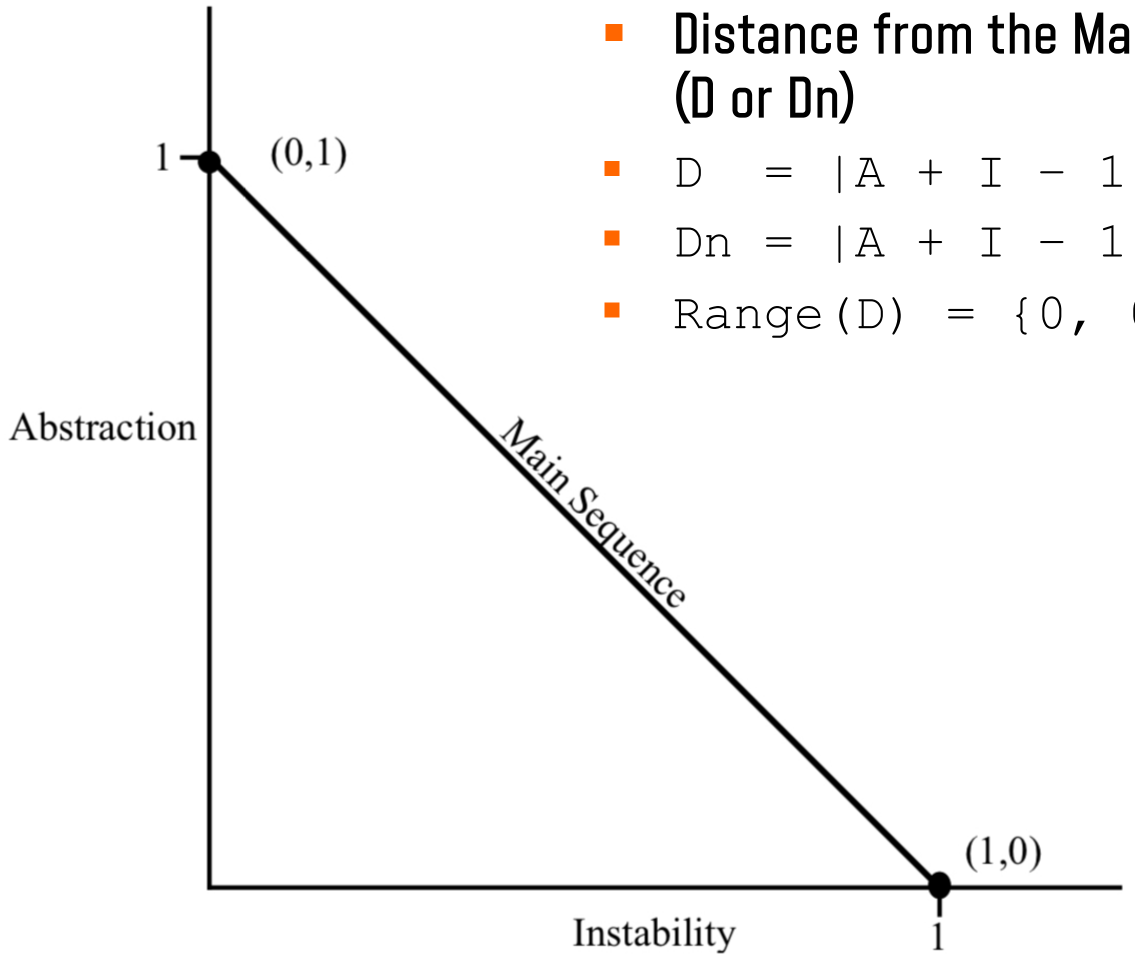
- `I = Ce / (Ce + Ca)`
- `Def(I) = {0, 1}`
- 0 corresponds to maximally stable and vice versa
- How to "use" Instability? Calculate average/mean instability for all modules in your system.
  - JDepend, NDepend, etc.



ABSTRACTNESS / EXTENSIBILITY

▪MAXIMALLY STABLE MODULES OPEN FOR MODIFICATION

- Measure of Open/Closed principle application on a specific (object oriented) design, but we can generalize…
- Abstract ~ outgoing, directed data (not control) flow
- Total = total # of connections to and from the module (incl. abstract)
- `A = #Abstract / #Total`
- `Range(A) = {0, 1}`
- Where do we want to be?



- Distance from the Main Sequence (D or Dn)
- `D  = |A + I - 1| / 2`
- `Dn = |A + I - 1|`
- `Range(D) = {0, 0.707}`

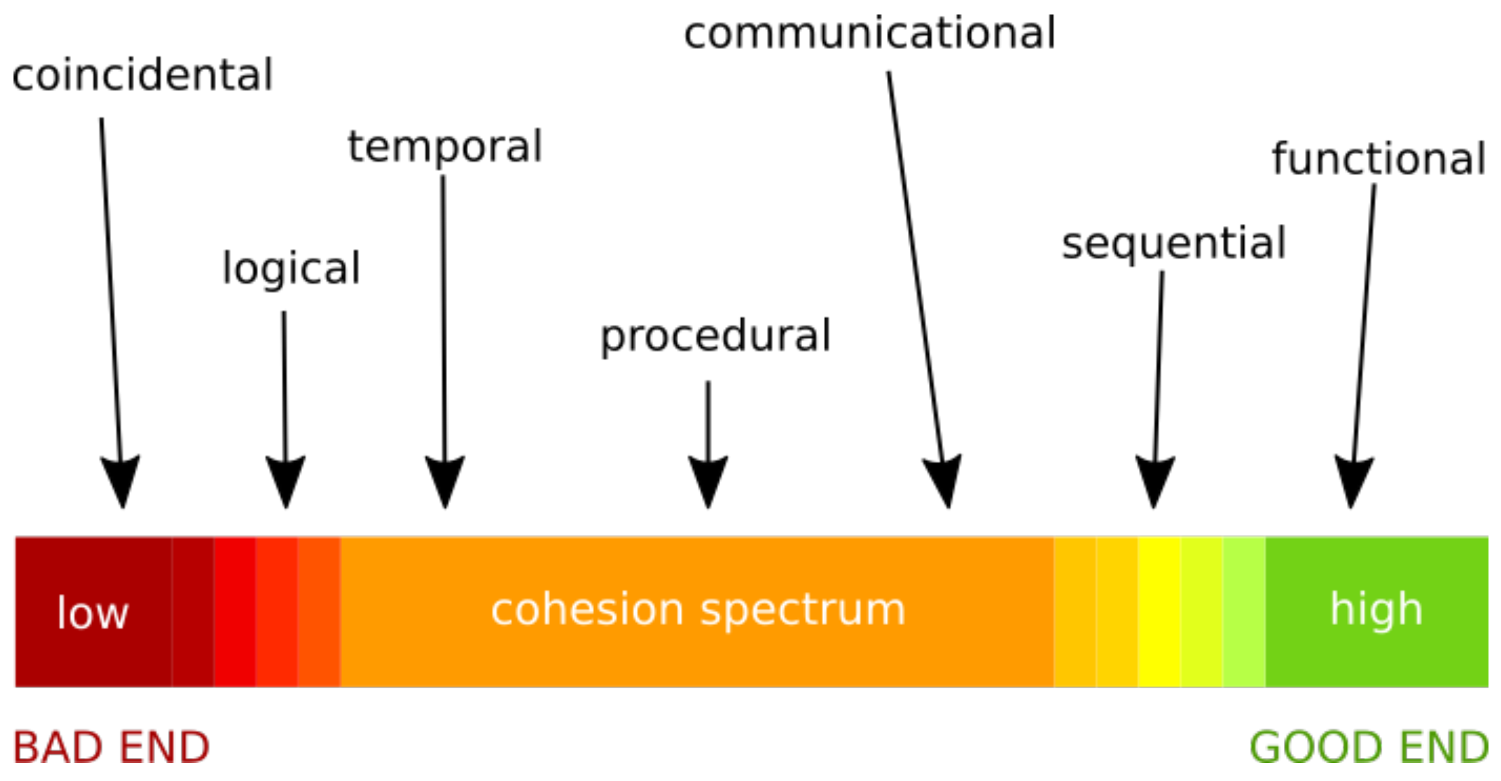
COHESION

WHAT'S IN THE BOX?

TYPES OF COHESION

WHY ARE "THINGS" TIED TOGETHER?

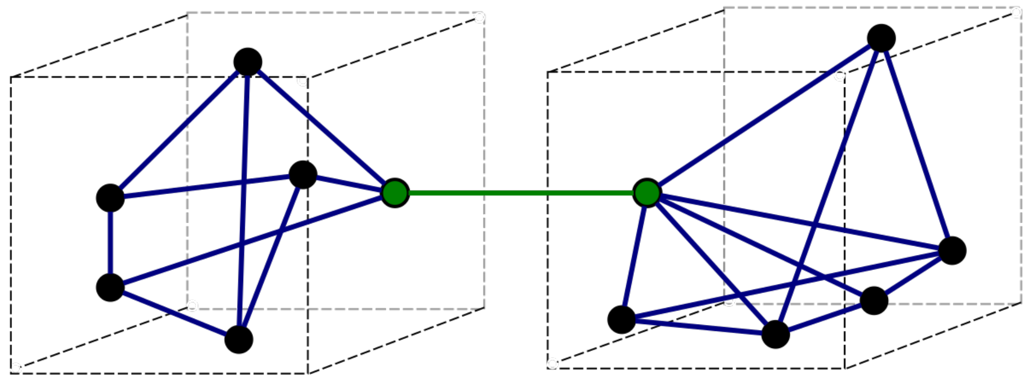
- Cohesion is a **property** of what's in the box.
- What is the impact of good/bad cohesion?

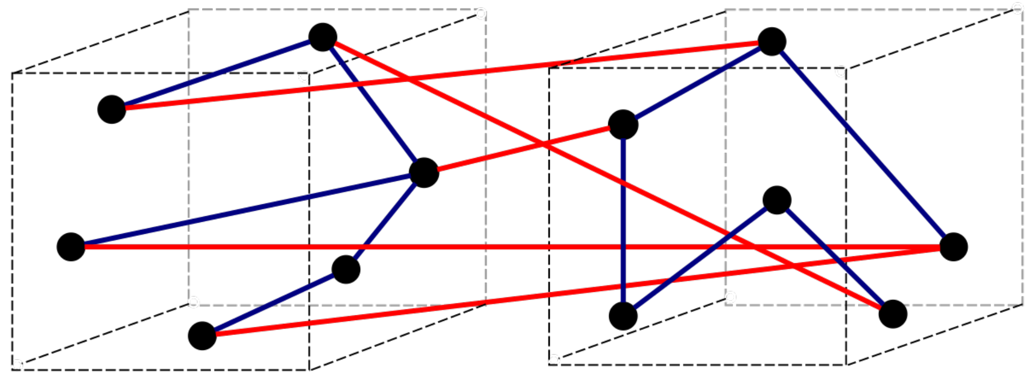


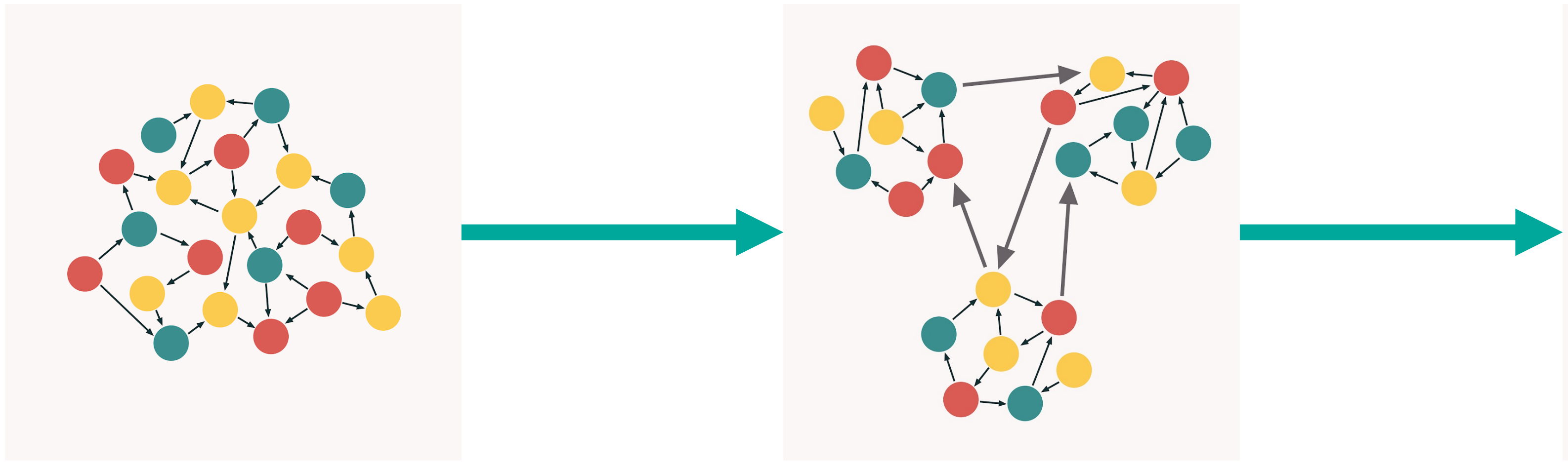
LOOSE COUPLING, HIGH COHESION

WHAT IS THE IMPACT OF GOOD/BAD COHESION?

Cohesion and coupling go hand in hand.


a) Good (loose coupling, high cohesion)


b) Bad (high coupling, low cohesion)



Dependencies.

Grouping.

Dependencies.

MODULARITY AND SOFTWARE COST

COST TO INTEGRATE VS. COST PER MODULE

WHAT TO REMEMBER?

DESIGN FOR FLOW OF DATA

- FLOW OF CONTROL IS EXTERNALIZED (TO THE INFRASTRUCTURE)
- AVOID FLOW OF CONTROL IN FAVOR OF MULTIPLE, SINGLE PURPOSE, DATA-CONNECTED MODULES

COMMODITY
OWN