BUSINESS BLUEPRINT DERIVATION

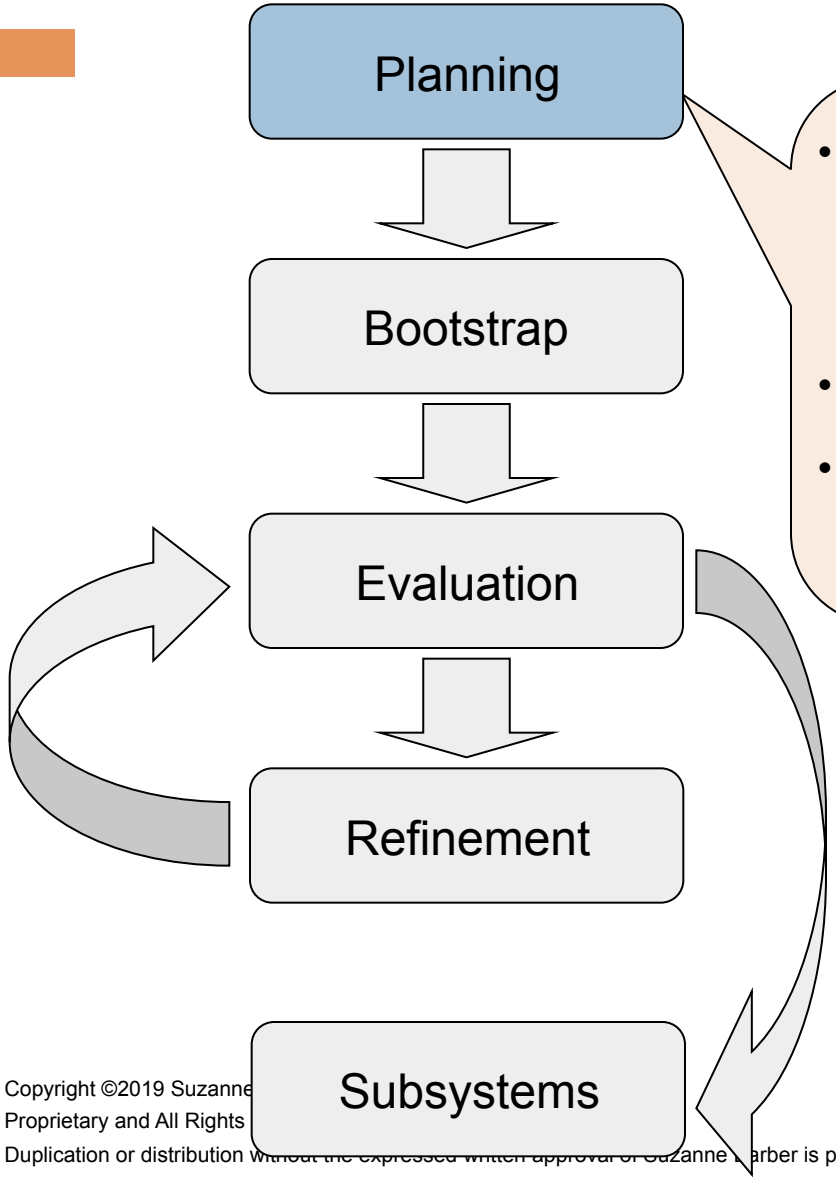
THE PROCESS

Suzanne Barber

Important Features of the BB (or any!) Derivation Process

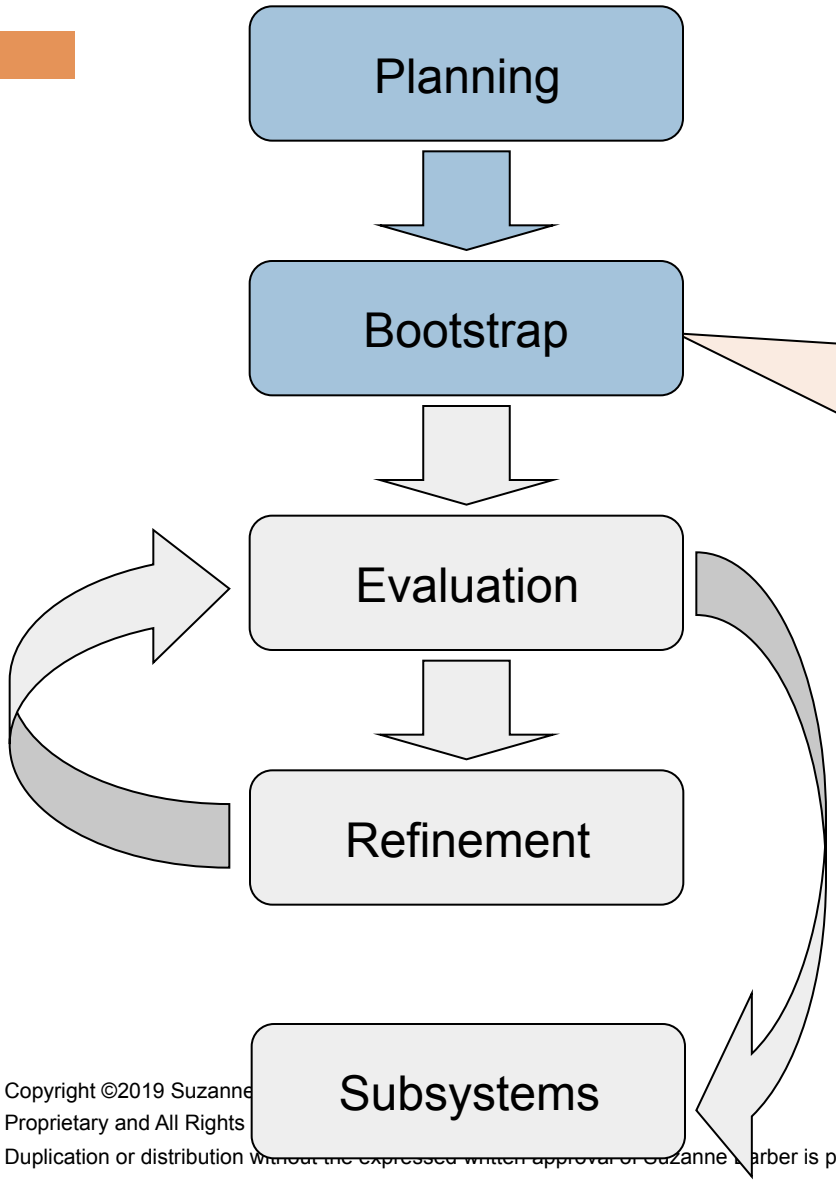
- Pre-Derivation Planning
  - Not ad hoc, but disciplined
  - Identify what the architecture should represent based on what stakeholders require
  - Determine a "plan of attack"
  - Establish clear metrics for success
- Continuous, Iterative refinement and evaluation
  - Can't derive architecture in one giant step – lots of individual decisions
    - No way to maintain traceability otherwise (why did I get the architecture I got?)
  - You (typically) have multiple options at each decision point, and few decisions are unequivocal, so you have to evaluate after each decision to ensure you're heading in the right direction
    - Somewhat analogous to coding and testing incrementally

Business Blueprint Derivation Process



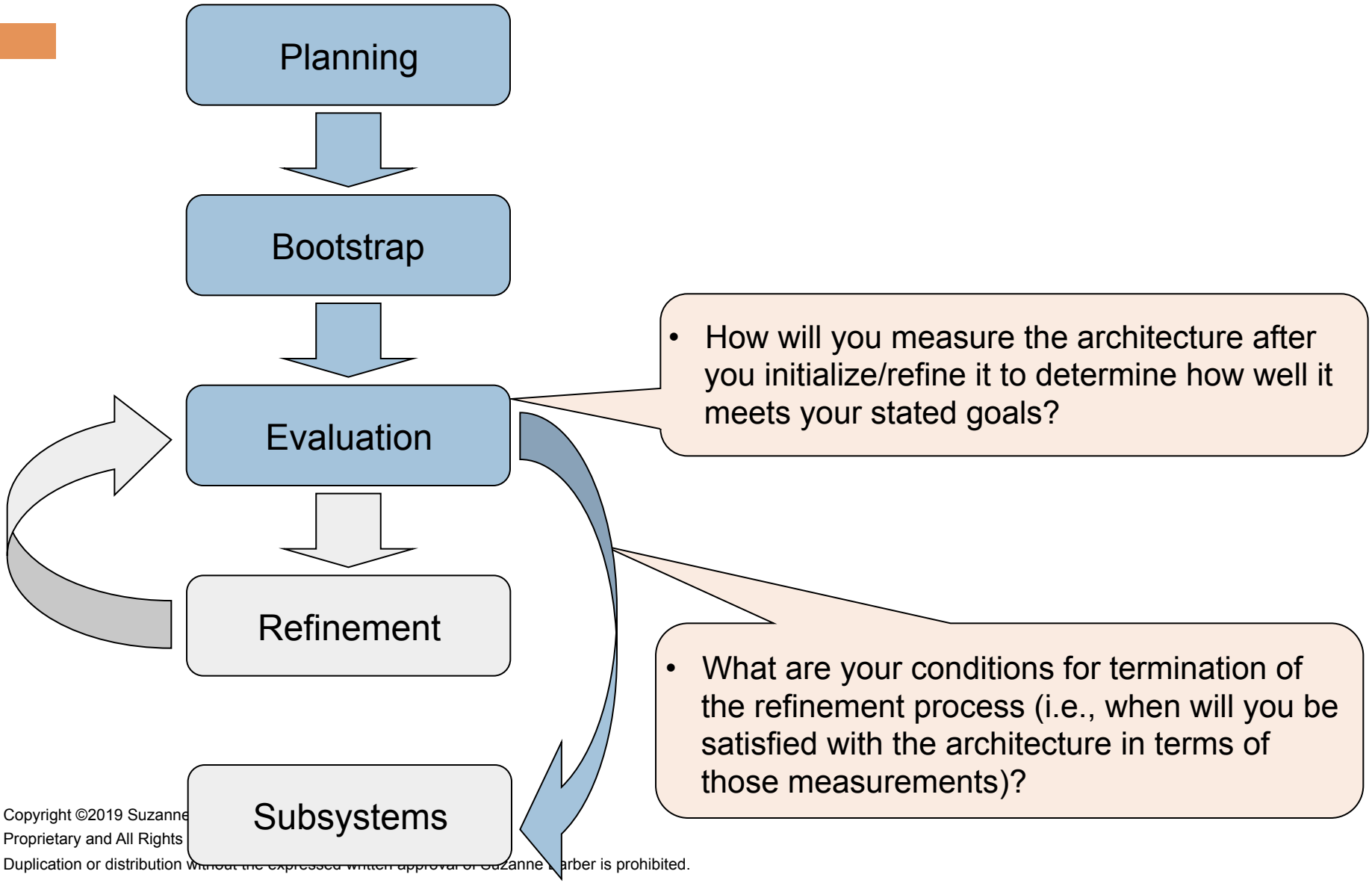
- What do you want your architecture to convey to stakeholders? (What are your architecture "goals" and how will you evaluate the architecture w.r.t. those goals?)
- Do you have the content you need?
- Which portion of that content should be included?

Business Blueprint Derivation Process

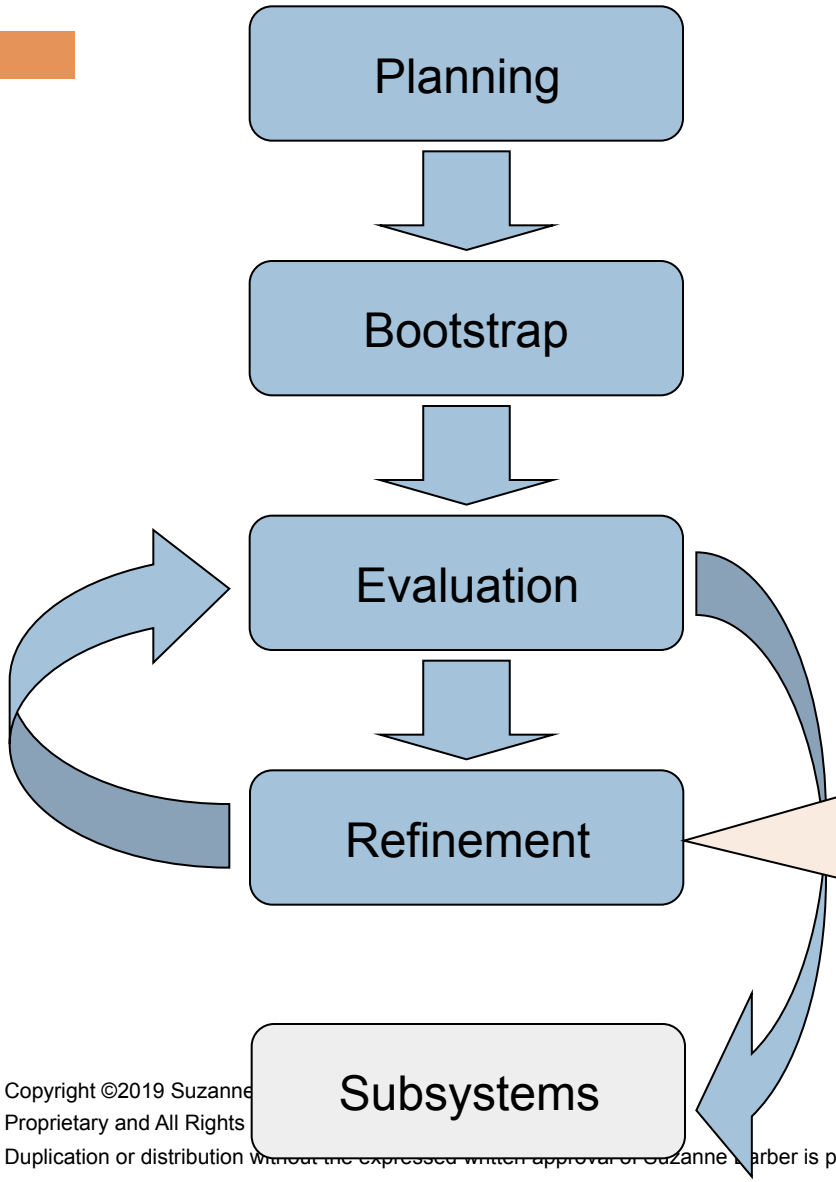


- When you have no architecture, what's the best approach for defining an initial architecture that will get you furthest toward your intended goals given the content you have available?

Business Blueprint Derivation Process

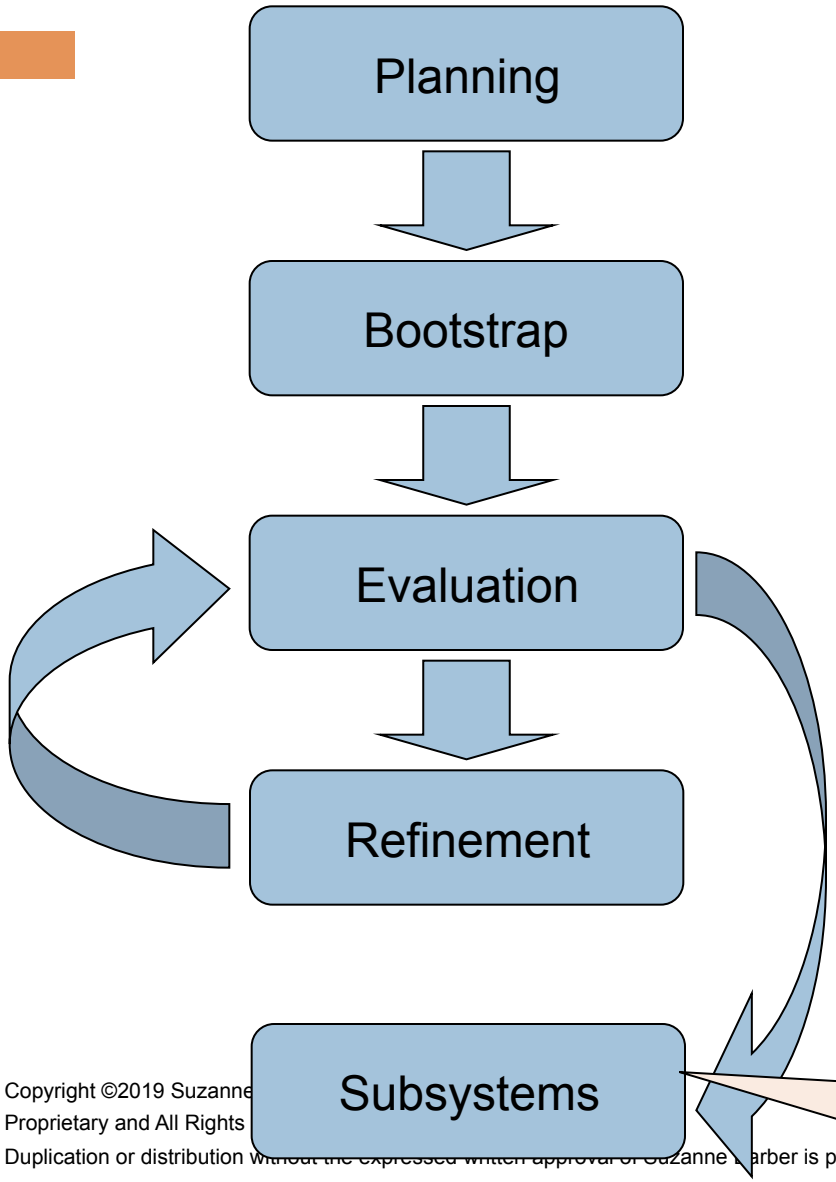


Business Blueprint Derivation Process



- Given an unsatisfactory evaluation, what is your plan for refining the architecture to better meet stated goals?
- Why is your refinement plan rational in terms of the specific measurements used to evaluate w.r.t. stated goals?

Business Blueprint Derivation Process



- Are there subsystems suggested by selected goals?

Announcements

- Final Exam Presentations
  - Fictional Company, Product and Situation
  - Recommend using the Beyond Bullet Points template
- Project Milestone #2 Rubric posted to FILES section

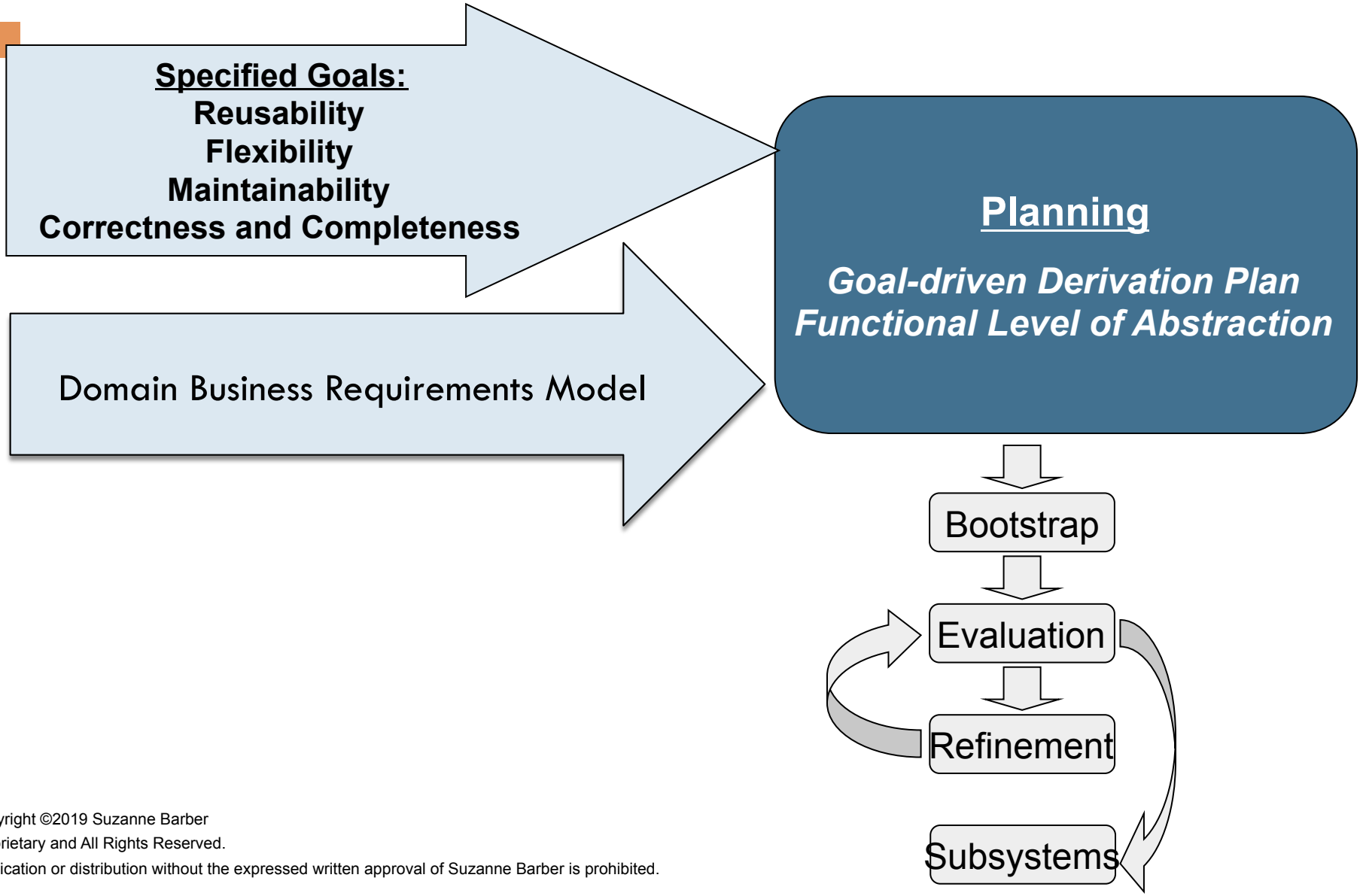
EXAMPLE ….

STEP-BY-STEP BUSINESS BLUEPRINT DERIVATION PROCESS

Pre-Derivation Planning

- What do you want your Business Blueprint to convey to stakeholders?
  - What qualities (goals) are important, and how do you intend to achieve them?
- Do you have the domain requirements content you need?
  - Requirements in Domain Models are consistent across stakeholders
  - What level of functional abstraction will you select?

In-Class Example: Specified Goals



Stakeholder goals (desired properties for the system)

- Per the stakeholder requirements, system goals are asserted with no priority given.
  - Reusability
    - Degree to which technologies adhering to the architecture can be reused
  - Flexibility
    - Ways in which technologies adhering to the architecture can be combined
  - Maintainability
    - Ease by which technologies adhering to the architecture can be maintained or extended
  - Completeness and Consistency
    - Necessary domain tasks, data, and events are present in the BB and represented correctly

For each goal, you need …

- An approach for achieving goal (heuristics and strategies)
- Metric(s) for evaluating Business Blueprint to demonstrate that you've achieved goal
- A means for resolving conflicts when goal conflicts arise

In-Class EXAMPLE PLAN: Plan for "Reusability" Goal

- Derivation Plan for achieving "Reusability" goal:
  - Heuristic: "Reduce coupling between technologies, thereby reducing entanglements and facilitating reusability"
    - Strategies for achieving heuristic:
      - Move services to BBCs where the services are used by the most other services (based on I/O)
      - Move attributes to BBCs where they are most used by services
  - Measurement Plan for evaluating BB w.r.t. "Reusability"
    - Coupling Metric
      - "Num Input/Output between BBCs"
        - For each BBC:
          - Num service inputs that come from another BBC + Num service outputs that go to another BBC

In-Class EXAMPLE PLAN: Plan for "Flexibility" Goal

- Derivation Plan for achieving "Flexibility" goal:
  - Heuristic: "Reduce number of services per technology to increase the number of possible technology combinations" (i.e., power set)
    - Strategies for achieving heuristic:
      - Split BBCs to reduce size
  - Measurement Plan for evaluating BB w.r.t. "Flexibility"
    - Size/Complexity Metric
      - "Num Services in BBC"
        - For each BBC:
          - Number of services in BBC
      - "Num Attributes in BBC"
        - For each BBC:
          - Number of attributes in BBC

In-Class EXAMPLE PLAN: Plan for "Maintainability" Goal

- Derivation Plan for achieving "Maintainability" goal:
  - Heuristic: "Reduce number of services per technology to simplify technology complexity"
    - Strategies for achieving heuristic:
      - Split BBCs to reduce size
  - Heuristic: "Reduce number of attributes per technology to simplify technology complexity"
    - Strategies for achieving heuristic:
      - Group data attributes based on composition
  - Heuristic: "Reduce coupling between technologies, thereby reducing interface complexity"
    - Strategies for achieving heuristic:
      - Move services to BBCs where to the BBCs where they are used by the most other services
      - Move attributes to BBCs where they are most used by services

In-Class EXAMPLE PLAN: (cont'd)
Plan for "Maintainability" Goal

- Measurement Plan for evaluating BB w.r.t. "Maintainability"
  - Size/Complexity Metric
    - "Num Services in BBC"
      - For each BBC:
        - Number of services in BBC
    - "Num Attributes in BBC"
      - For each BBC:
        - Number of attributes in BBC
  - Coupling Metric
    - "Num Input/Output between BBCs"
      - For each BBC:
        - Num service inputs that come from another BBC + Num service outputs that go to another BBC

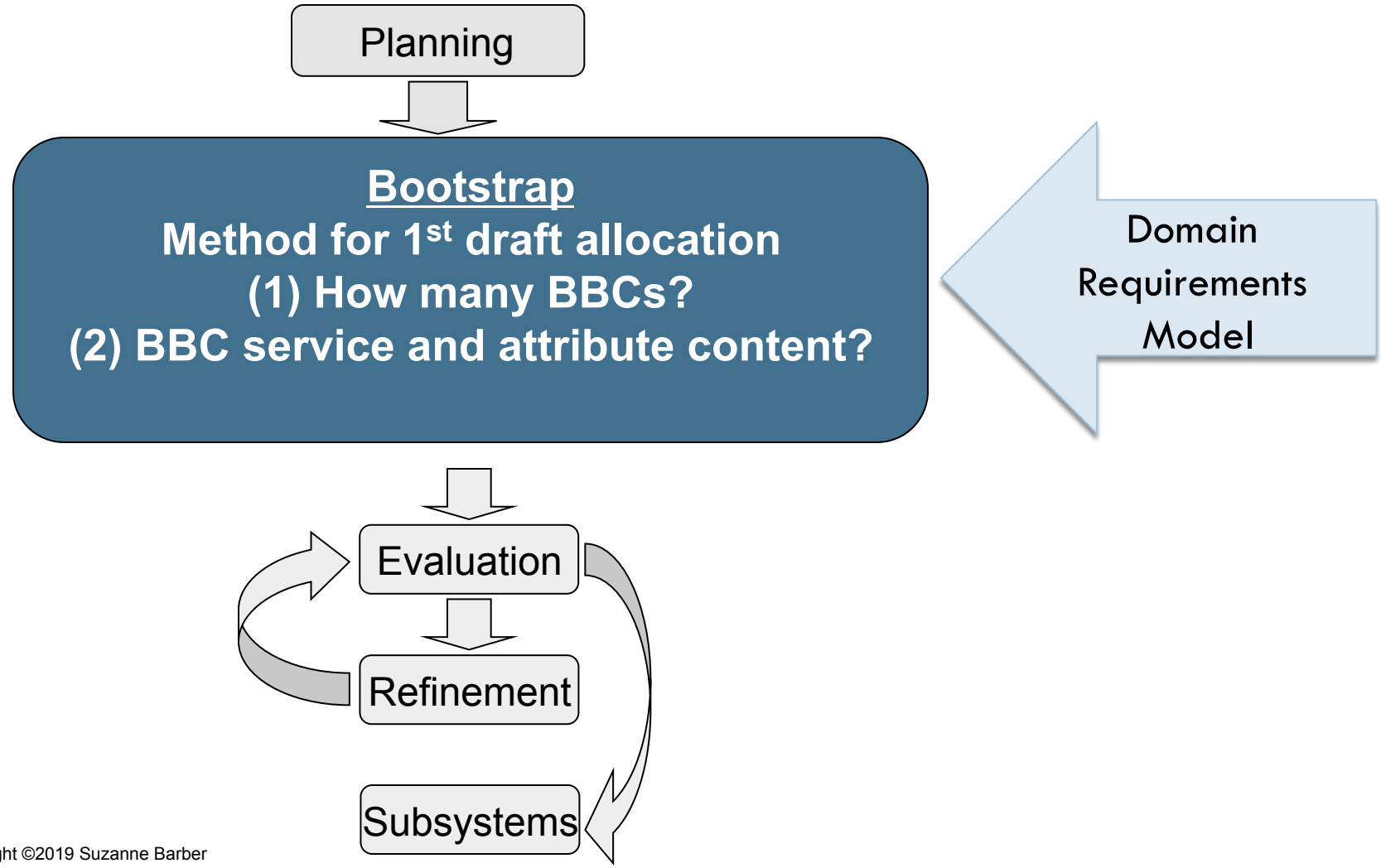
In-Class EXAMPLE PLAN:
Plan for "Completeness and Consistency" Goal

- Derivation Plan for achieving "Completeness and Consistency" goal:
  - Heuristic: "All necessary tasks, data, and events should be represented in the BB"
    - Strategies for achieving heuristic:
      - Add attributes referenced in task I/O but not defined
  - Heuristic: "A given task, data, or event should be provided by one and only on BBC"
    - Strategies for achieving heuristic:
      - Eliminate duplicate services in two BBCs by moving to new BBC associated with common parent
      - Eliminated duplicate attribute by moving to BBC that it is needed by the most services
        - Number of attributes defined in more than one BBC

In-Class EXAMPLE PLAN: (cont'd)
Plan for "Completeness and Consistency" Goal

- Measurement Plan for evaluating BB w.r.t. "Completeness and Consistency"
  - Completeness Metrics
    - "Num missing services"
      - For the BB as a whole
        - Number of tasks selected from the domain model but not defined in any BBC
    - "Num missing attributes"
      - For the BB as a whole
        - Number of data or events used defined in BB service I/O but not defined as an attribute
  - Duplication Metric
    - "Num Duplicate Services"
      - For the BB as a whole
        - Number of services defined in more than one BBC
    - "Num Duplicate Attributes"
      - For the BB as a whole
        - Number of attributes defined in more than one BBC

BB Derivation Process



Domain Requirements – Inputs to the Business Blueprint derivation process

Task Template

Basic Task Information

Task Name:
Task Description:
Subtasks (zero or more):
Execution Duration:
Execution Frequency:

Task Performance Information

Location where task performed: (one or more)
Performers: Personnel Roles or Technology that can perform task: (one or more)

Task Input and Output,

Data/Event Input (one or more):
Input Name (e.g. DataInput1):
Data/Event Name:
Cardinality (Data only):
Sender of Input:
Task producing Output or EXTERNAL:
Task Performer:

Data/Event Output (one or more):
Output Name:
Data/Event Name:
Cardinality (Data only):
Receiver of Output:
Task requiring Input or EXTERNAL:
Task Performer:

Resources Required (zero or more):

Personnel:
Equipment:

Task Timing

Pre-condition Statement: Example …

1. _DataInput1 Sent AND DataInput2 Sent__ OR
2. _NumPatients < 5 AND TaskX-Started OR
3. _____ OR ...

Post-condition Statement

1. _PatientRecord Stored_____ OR
2. _TaskX-Completed_____ OR
3. _____ OR ...

In-Class EXAMPLE: Task Templates

- Task *T1*
  - Performed By: *P1*
  - At Location: *L1*
  - Used in Use Case(s): *UP1*
  - Inputs:
    - Event *E3* from *External*
  - Outputs:
    - Event *E1* to task *T2*
    - Data *D1* to task *T2*

- Task *T2*
  - Performed by: *P1*
  - At Location: *L1*
  - Used in Use Case(s): *UP1*
  - Inputs:
    - Event *E1* from task *T1*
    - Data *D1* from task *T1*
  - Outputs:
    - Event *E2* to task *T3*

In-Class EXAMPLE: Task Templates (cont'd)

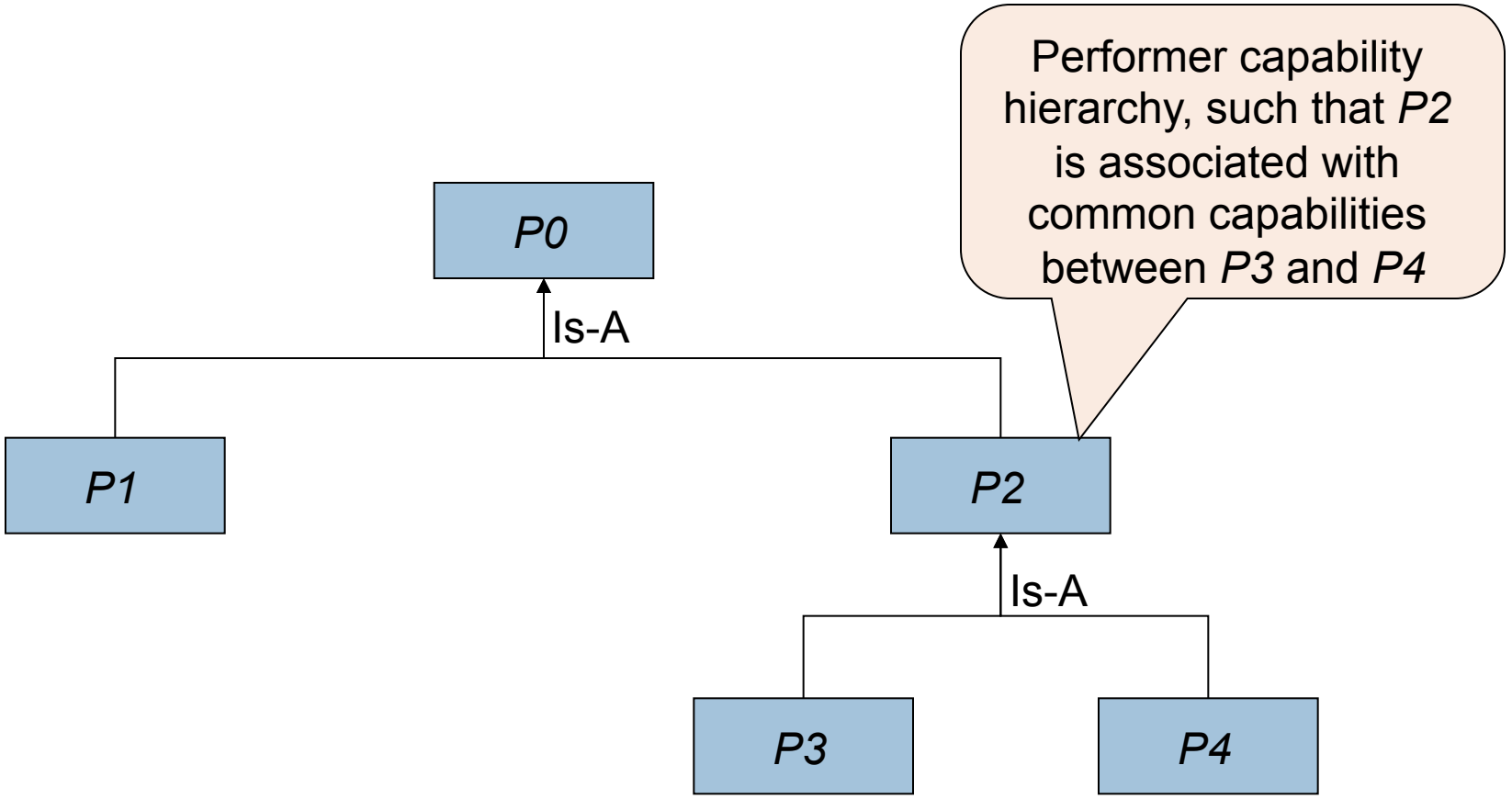
- Task *T3*
  - Performed By: *P1*
  - At Location: *L1*
  - Used in Use Case(s): *UP2*
  - Inputs:
    - Event *E4* from *External*
    - Event *E2* from task *T2*
    - Data *D3* from task *T4*
  - Outputs:
    - Data *D2* to task *T4*

- Task *T4*
  - Performed by: *P3*
  - At Location: *L1*
  - Used in Use Case(s): *UP2*
  - Inputs:
    - Data *D2* from task *T3*
    - Data *D4* from task *T5*
  - Outputs:
    - Data *D3* to task *T3*
    - Data *D2* to task *T5*

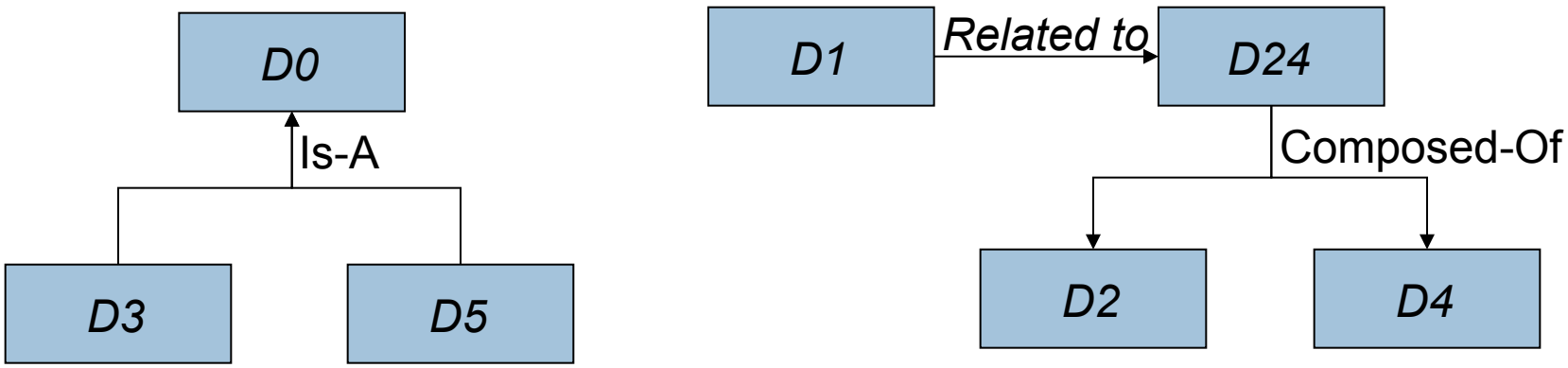
In-Class EXAMPLE: Task Templates (cont'd)

- Task *T5*
  - Performed by: *P3,P4*
  - At Location: *L2*
  - Used in Use Case(s): *UP2*
  - Inputs:
    - Data *D2* from task *T4*
  - Outputs:
    - Data *D4* to task *T4*

In-Class EXAMPLE: Solution Hierarchy



In-Class EXAMPLE: Entity-Relationship Data Diagram

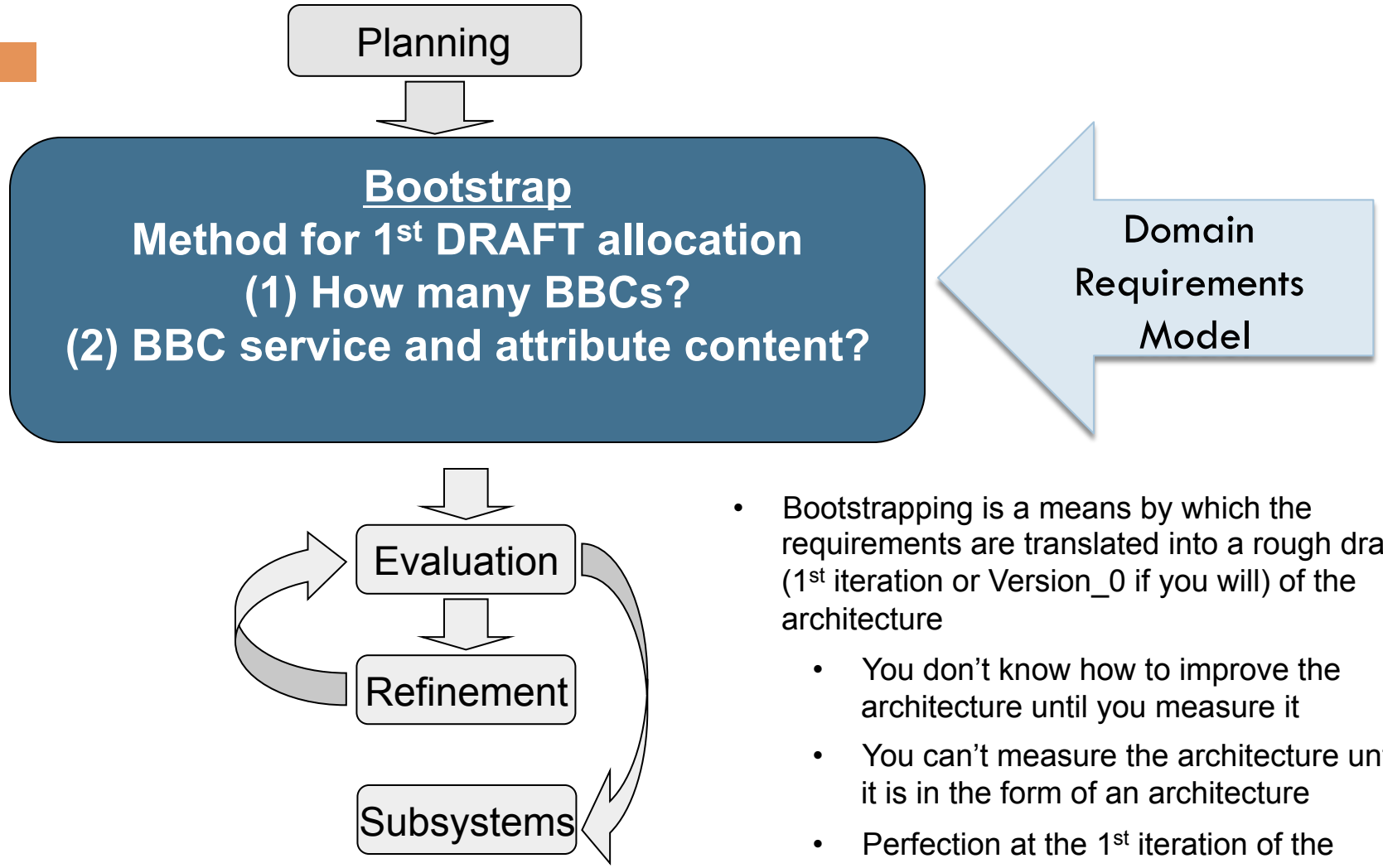


In-Class EXAMPLE: Use Cases

- Use Case UC1
  1. Task *T1*
  2. Task *T2*

- Use Case *UC2*
  1. Task *T3*
  2. Task *T4*
  3. Task *T5*

BB Derivation Process



- Bootstrapping is a means by which the requirements are translated into a rough draft (1st iteration or Version_0 if you will) of the architecture
  - You don't know how to improve the architecture until you measure it
  - You can't measure the architecture until it is in the form of an architecture
  - Perfection at the 1st iteration of the architecture is not possible so the bootstrap version aims to set up the best possible starting point.

3 Bootstrapping Options … Pick one or select another.

- Performers
  - Possible Rationale: "A performer executes a collection of tasks in part because those tasks require some capability the performer possesses. Thus, technologies should be matched to those tasks needing the capabilities the technologies can offer."
- Locations
  - Possible Rationale: "Spatial Locality – Tasks performed at a given location should be performed by the same technology to avoid the need for mobility."
- Usage Cases
  - Possible Rationale: "Temporal Locality – Tasks often performed consecutively should be performed by the same technology to reduce overhead associated with switching technologies when transitioning between tasks."

BOOTSTRAPPING BASED ON PERFORMERS

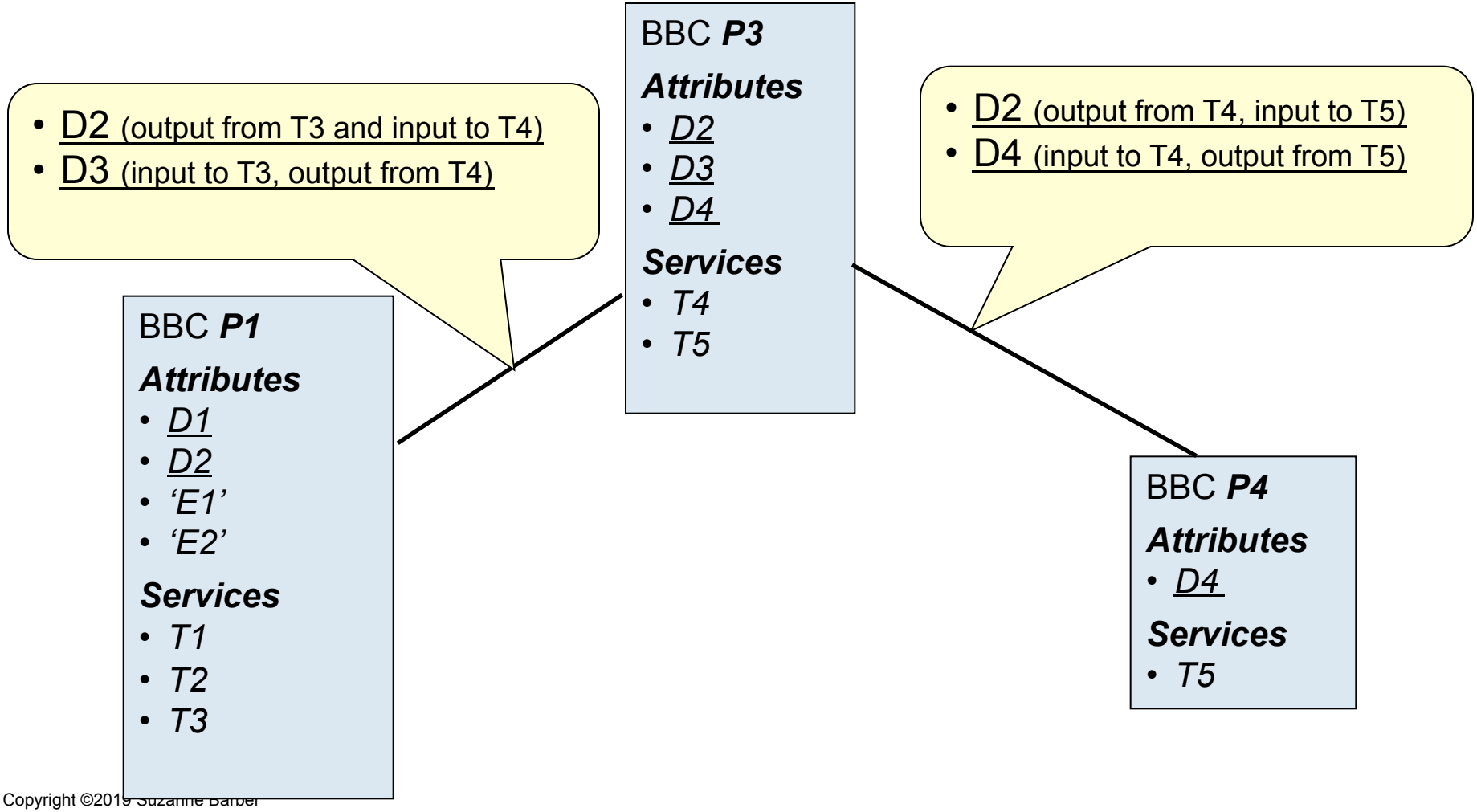
Initial Declarative Model (Service and Attribute Assignments)

Initial Integration Model (BBC dependencies)

BBC	Service	Output	Input	Received From
P1	*T1*	*E1*	*E3*	*External*
		D1		
	T2	*E2*	*E1*	*T1*
			D1	*T1*
		D2	*E4*	*External*
			E2	*T2*
			D3	*T4*
P3	*T4*	*D3*	*D2*	*T3*
		D2	*D4*	*T5*
	T5	*D4*	*D2*	*T4*
P4	*T5*	*D4*	*D2*	*T4*

Begin with lowest level in performer hierarchy

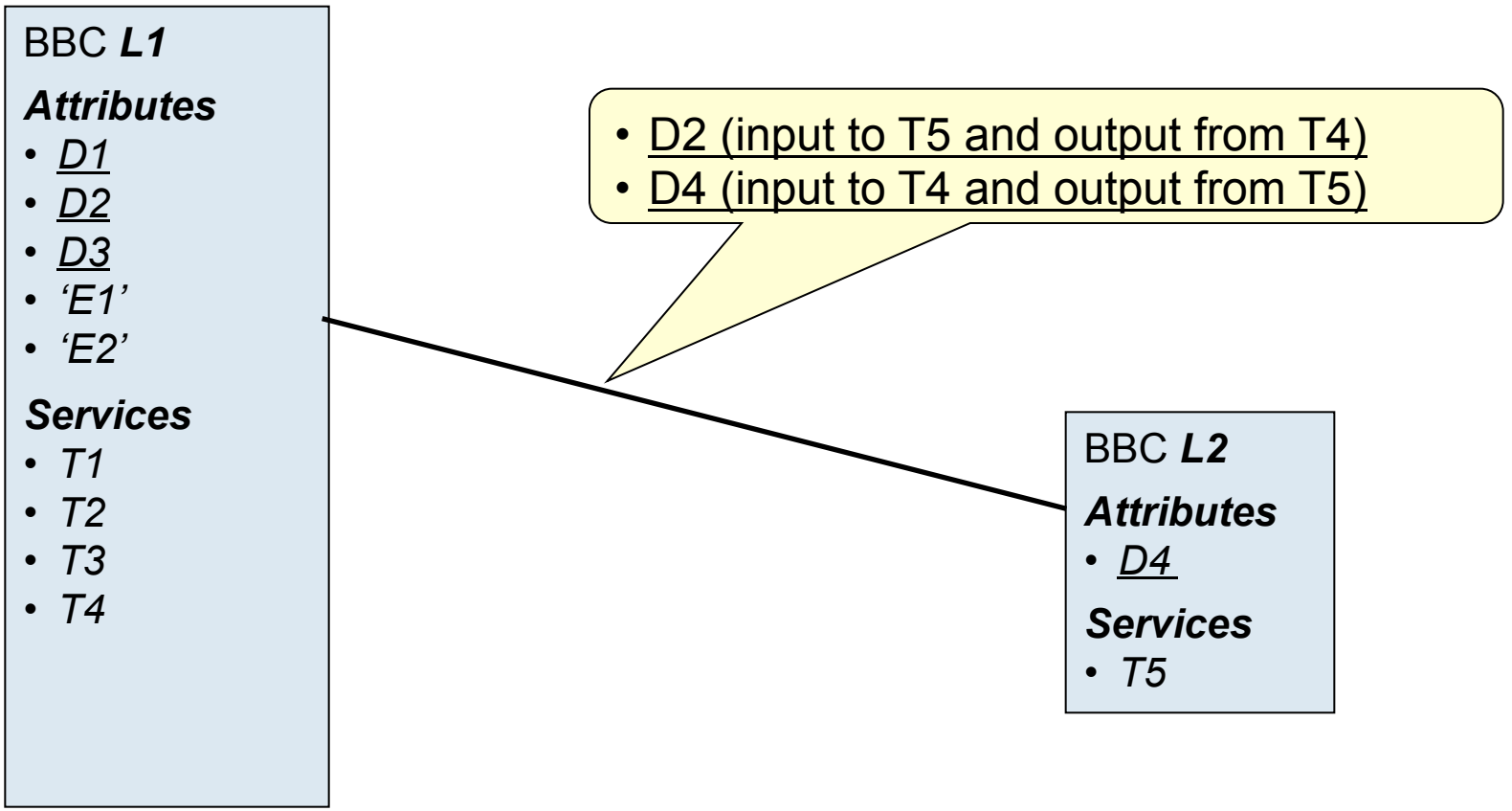
Bootstrapping using Performers
Graphical Depiction of Resulting Business Blueprint



Bootstrapping using Locations
Creating the Responsibility Spreadsheet

BBC	Service	Output	Input	Received From
L1	*T1*	*E1*	*E3*	*External*
		D1		
	T2	*E2*	*E1*	*T1*
			D1	*T1*
	T3	*D2*	*E4*	*External*
			E2	*T2*
			D3	*T4*
	T4	*D3*	*D2*	*T3*
		D2	*D4*	*T5*
L2	*T5*	*D4*	*D2*	*T4*

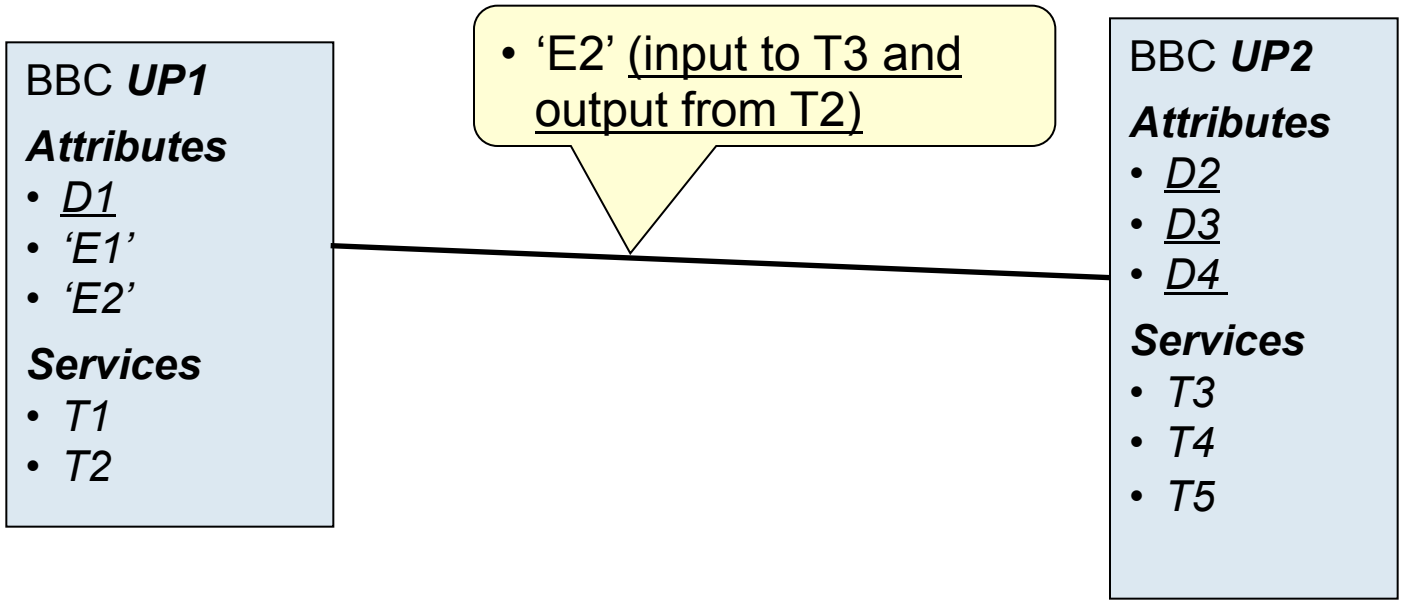
Bootstrapping using Locations
Graphical Depiction of Resulting Business Blueprint



Bootstrapping using Usage Profiles
Creating the Responsibility Spreadsheet

BBC	Service	Output	Input	Received From
UP1	*T1*	*E1*	*E3*	*External*
		D1		
	T2	*E2*	*E1*	*T1*
			D1	*T1*
UP2	*T3*	*D2*	*E4*	*External*
			E2	*T2*
			D3	*T4*
	T4	*D3*	*D2*	*T3*
		D2	*D4*	*T5*
	T5	*D4*	*D2*	*T4*

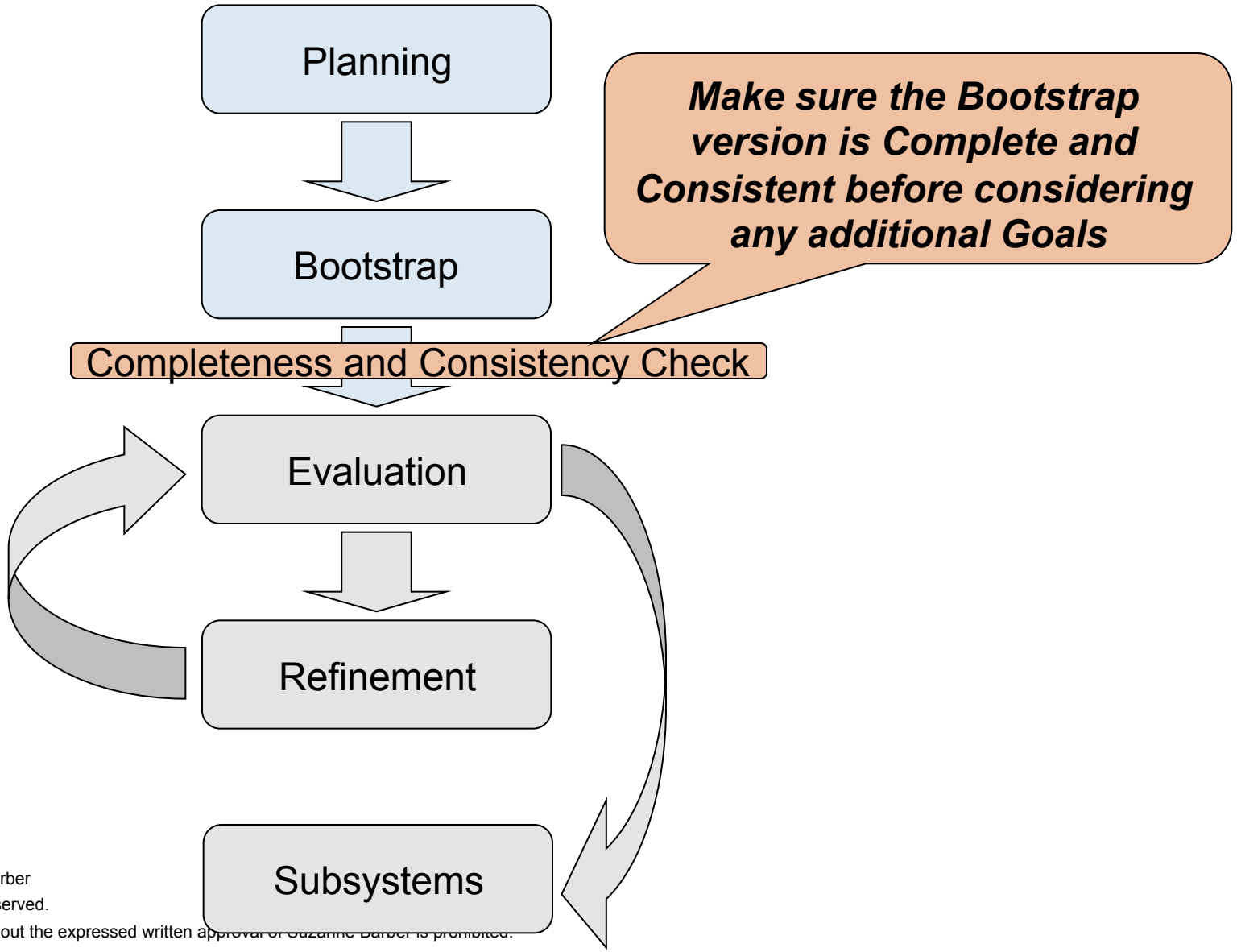
Bootstrapping using Usage Profiles
Graphical Depiction of Resulting Business Blueprint



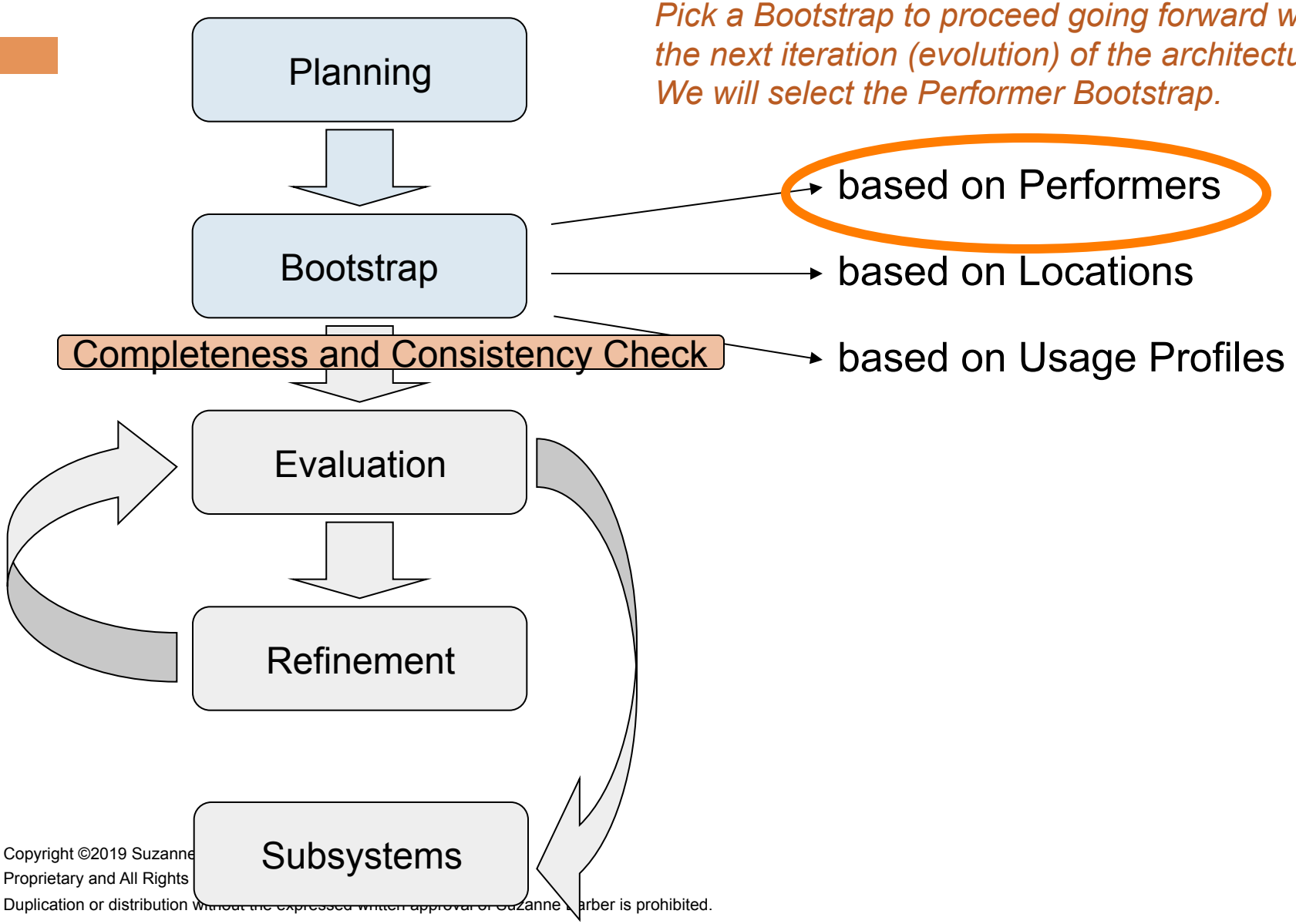
TOPIC EXERCISE

AFTER THE BOOTSTRAP… WHAT IS NEXT?

Business Blueprint Derivation Process

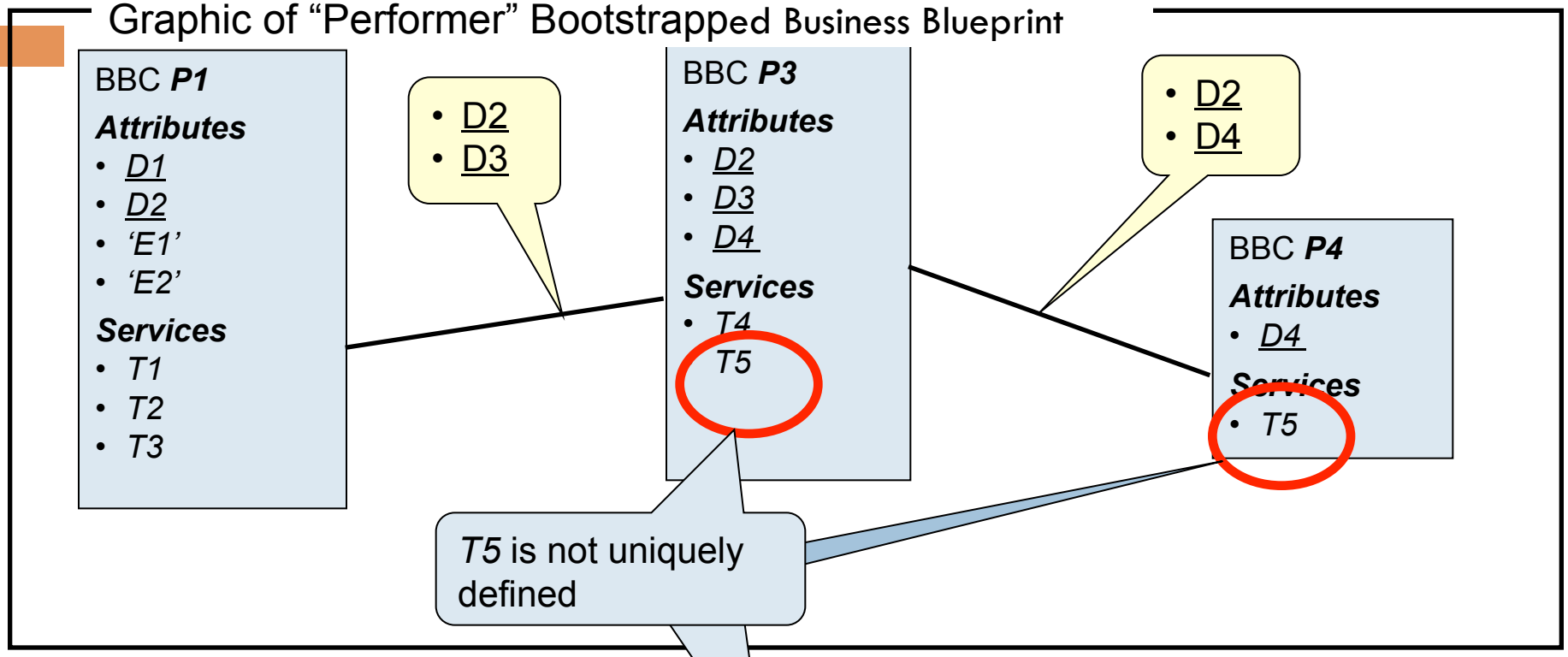


In-Class EXAMPLE: BB Derivation Process



Pick a Bootstrap to proceed going forward with the next iteration (evolution) of the architecture. We will select the Performer Bootstrap.

In-Class EXAMPLE: Evaluating "Completeness and Consistency" Detecting a Duplicate Services Error



Addressing the Duplicate Services Error

- Duplication of Services across BBCs
- Target: Make unique assignment of services to maximize cohesion (self sufficiency) of services

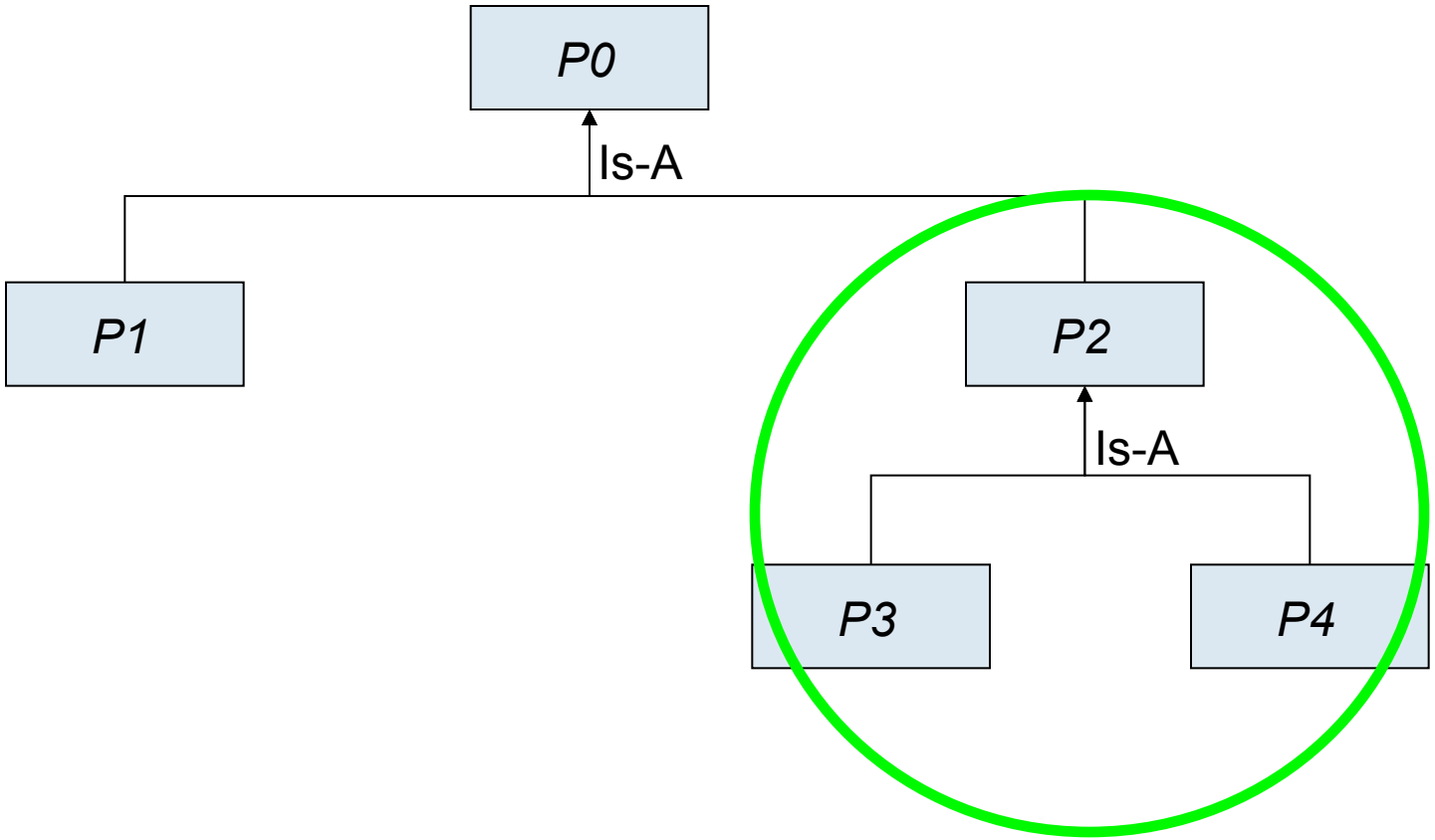
Addressing the Duplicate Services Error

- For BBCs owning the same service --
  - Do Roles (associated with all BBCs) have same parents in role concept hierarchy?
    - Yes?
      - Create new BBC = parent in role concept hierarchy
      - Assign responsibility for service and associated service output to new parent role BBC
        - If any BBCs are left empty after new assignments, then delete
  - Does each BBC own all the data/events required by service (input and pre-conditions)
    - Only one BBC? this BBC is assigned service and its associated output
    - More than one? architect's choice
    - None? BBC with the highest ratio value receives responsibility for service and associated output

$$\frac{\text{\# data/event elements (BBC Attributes) owned by BBC AND required by respective service}}{\text{\# TOTAL data/event elements required by respective service,}}$$

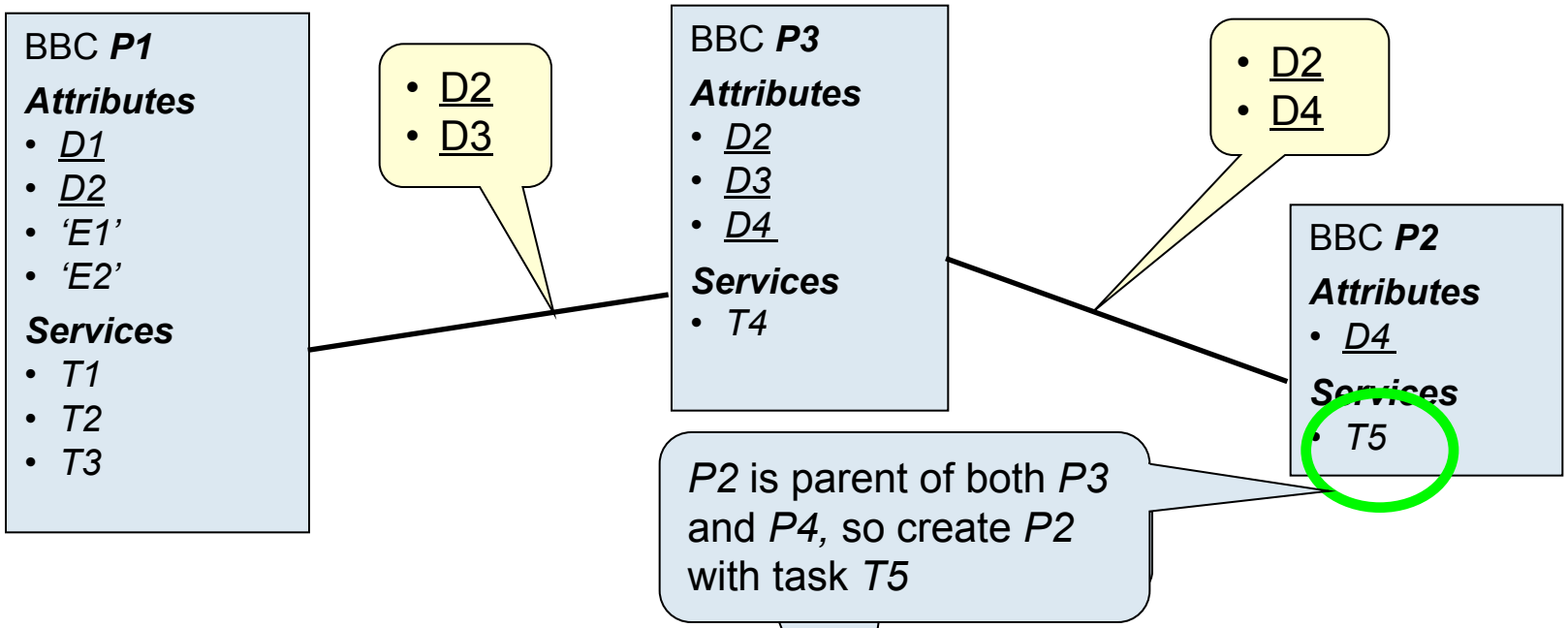
i.e. data/events required by services in input and check in precondition

Addressing the Duplicate Services Error using the Performer Hierarchy



In-Class EXAMPLE: Applying the strategy to correct the Duplicate Services Error

Graphic of BB with Duplicate Services Error corrected

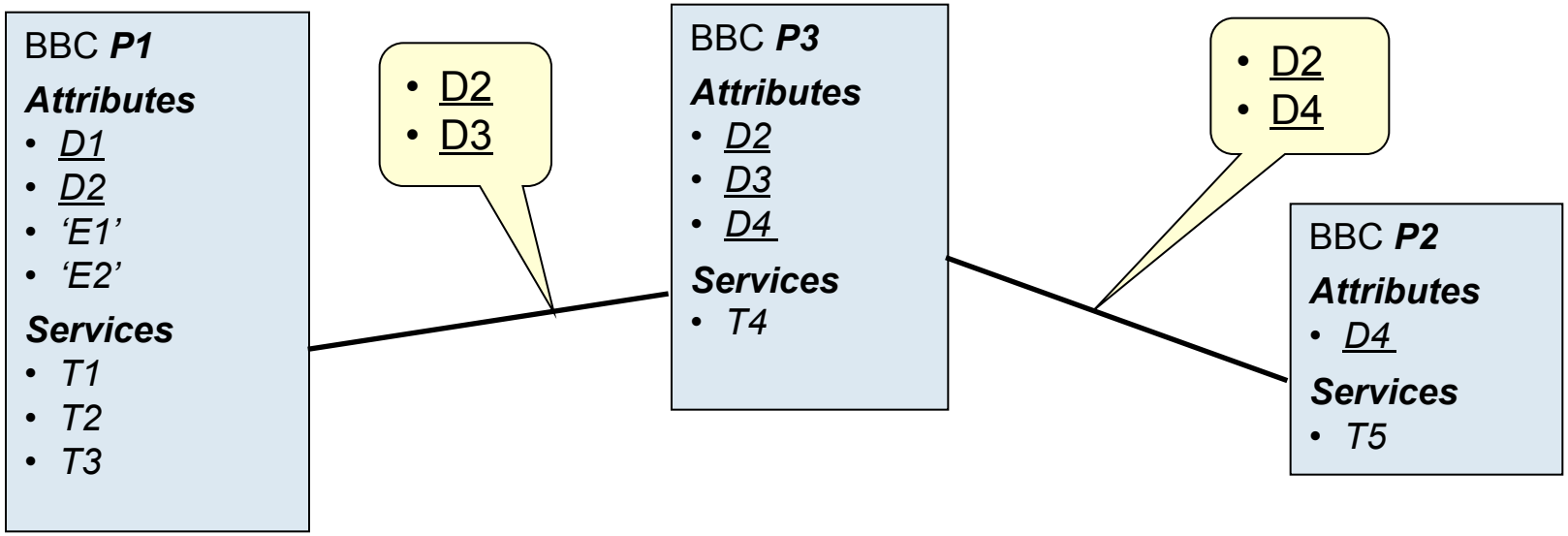


Strategy selected to correct Duplicate Services Error

- Create new BBC = parent in role concept hierarchy
  (i.e., P2 is parent of P3 and P4, and both P3 and P4 can perform T5)

In-Class EXAMPLE: Evaluating the BB after correcting the Duplicate Services Error

Graphic of BB with Duplicate Services Error corrected

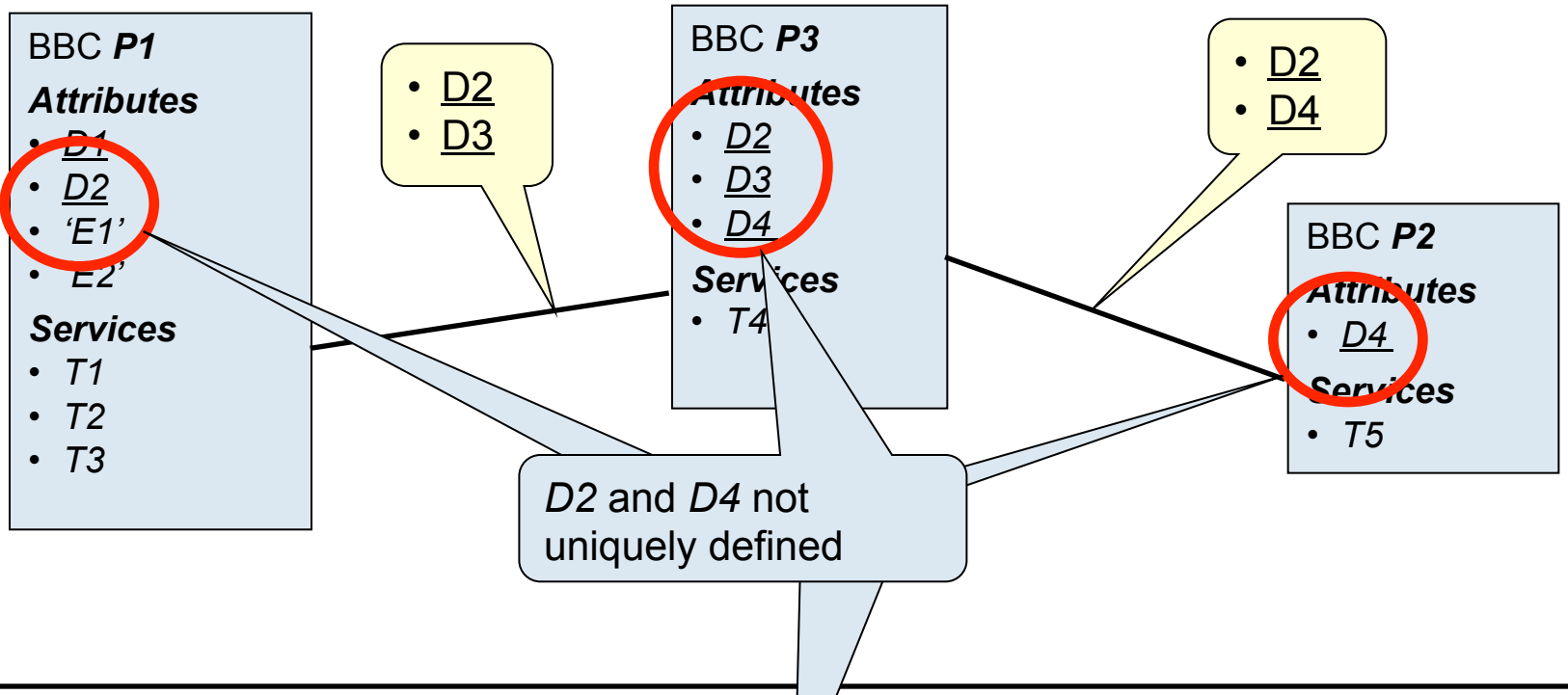


Relevant Metrics indicating no Duplicate Services Error

- Num_Duplicate_Services (BB) = 0

In-Class EXAMPLE: Evaluating "Completeness and Consistency" Detecting a Duplicate Attribute Error

Graphic of refined BB after correcting duplicate services



Relevant Metrics to detect Duplicate Attribute Error

- Num_Duplicate_Attributes(BB) = 2

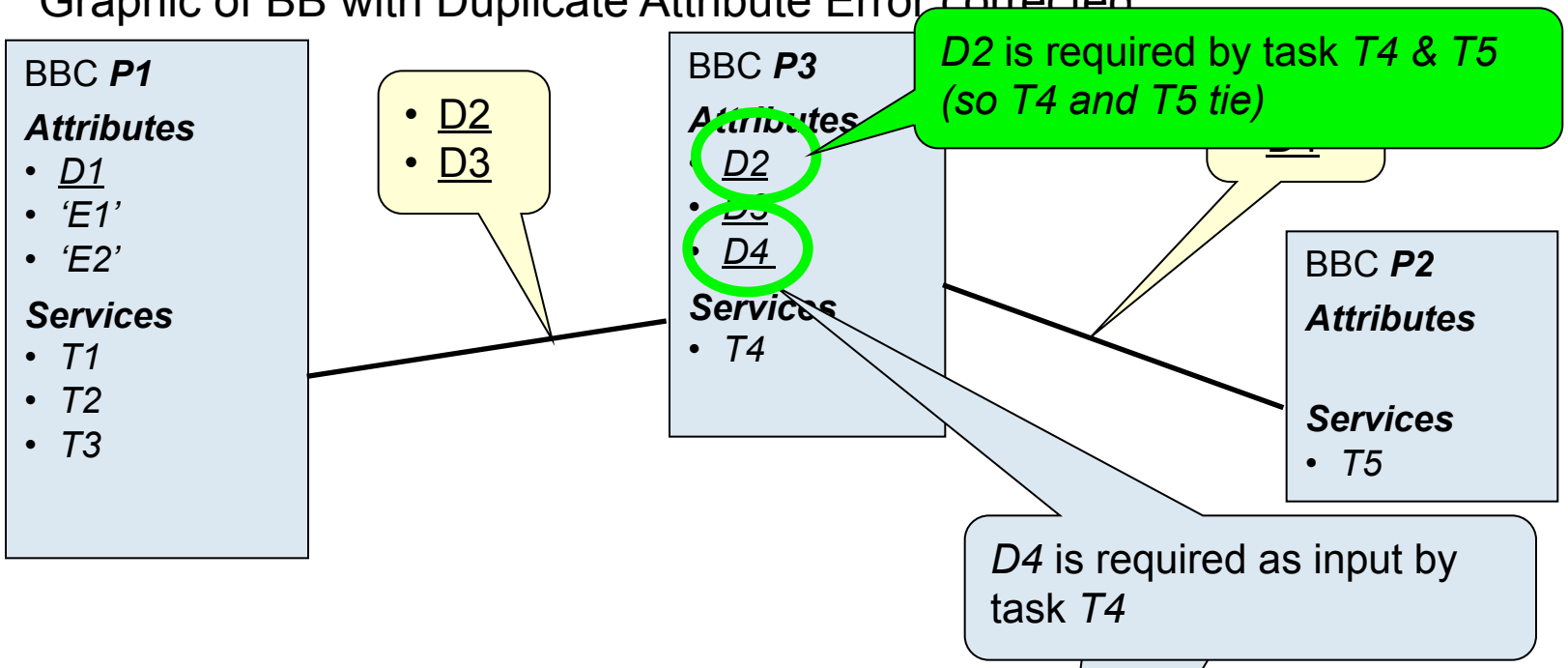
Addressing the Duplicate Attribute Error

- Duplication of Data/Events across BBCs (BBC Attributes)
  - Remember, Service Output assigned as BBC Attributes
  - Duplication due to Data/Events generated by more than one service
- Target: Make unique assignment of Attributes
  - Result may be
    - minimized coupling of services across BBCs OR
    - Increased coupling of services across BBCs
- Assignment of responsibility for data/event Attribute
  - Given to BBC owning the highest # of services requiring the respective data/event as input or part of pre-condition

In-Class EXAMPLE: Addressing the Duplicate Attribute Error

by moving attributes to BBCs where most demanded by services

Graphic of BB with Duplicate Attribute Error corrected

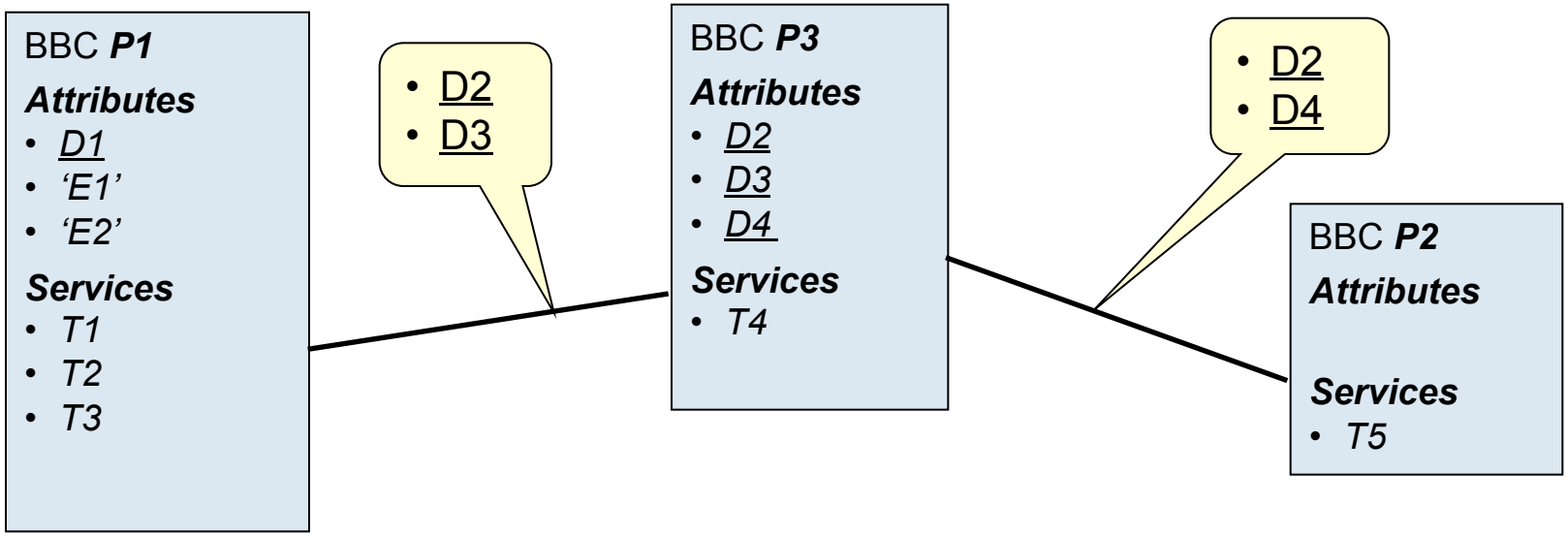


Strategy selected to correct Duplicate Attribute Err

- Given to BBC owning the highest # of services requiring the respective data/event as input or part of pre-condition (i.e., D4 is used by only T4 and D2 is used by T4 and T5, so arbitrarily associate with T4)

In-Class EXAMPLE: Evaluating the BB after correcting the Duplicate Attribute Error

Graphic of BB with Duplicate Attribute Error corrected

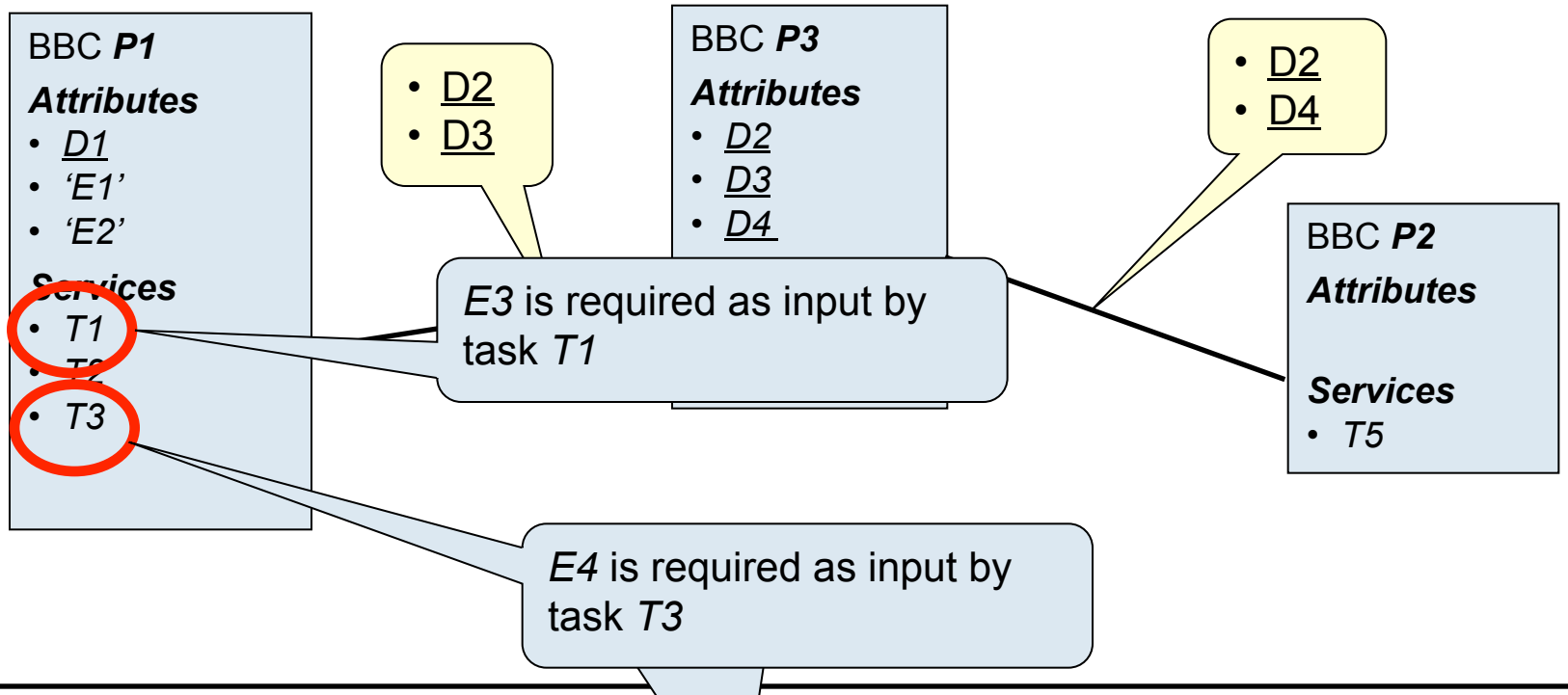


Relevant Metrics indicating no Duplicate Attribute Error

- Num_Duplicate_Attributes(BB) = 0

In-Class EXAMPLE: Evaluating "Completeness and Consistency" Detectina a Missina Attribute Error

Graphic of refined BB after correcting duplicate attributes



Relevant Metrics to detect Miss Attribute Error

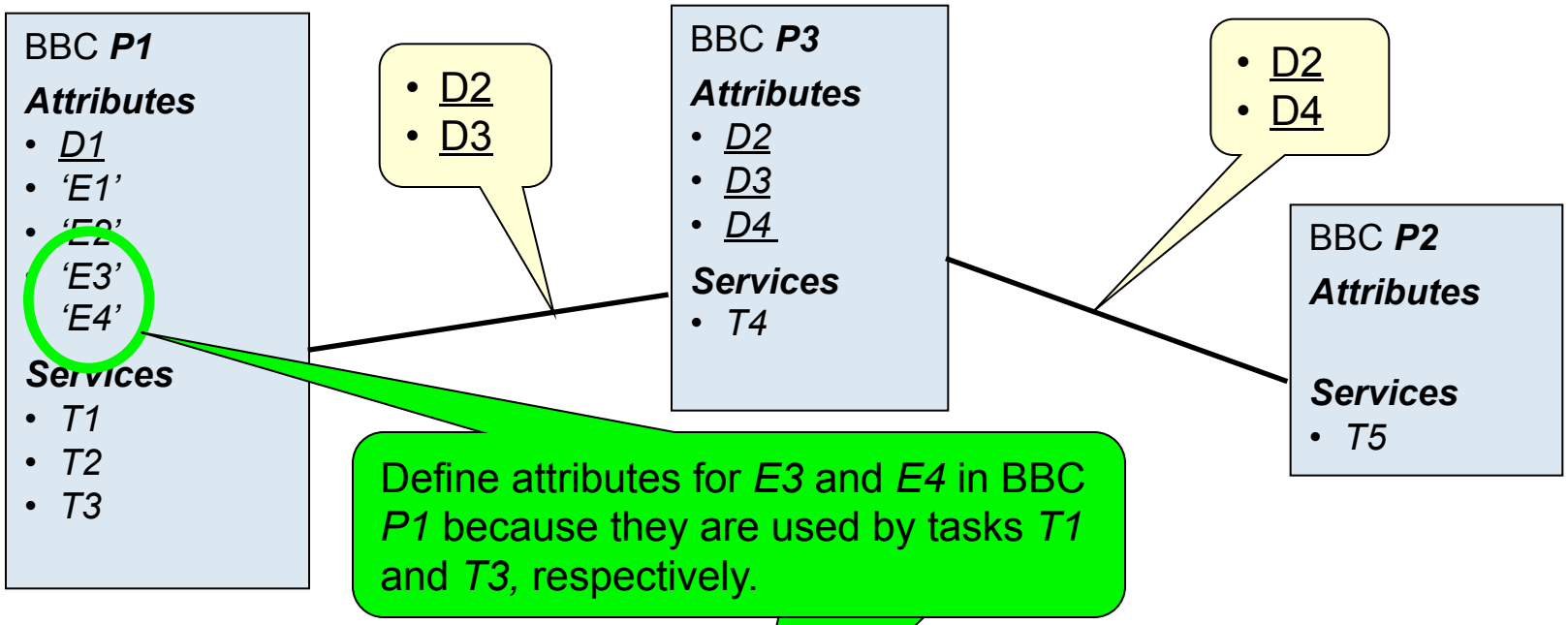
- Num_Missing_Attributes(BB) = 2

Addressing the Missing Attribute Error

- Only one service requires Input?
  - Assign data/event as Attribute to BBC that received responsibility for respective service requiring attributes as input
- More than one service requires Input?
  1. Assign data/event as Attribute to BBC with the highest number of services requiring data/event as input
  2. IF 1. Is not differentiating, architect's choice

In-Class EXAMPLE: Addressing the Missing Attribute Error by adding to BBCs where respective services located

Graphic of BB with Missing Attribute Error corrected

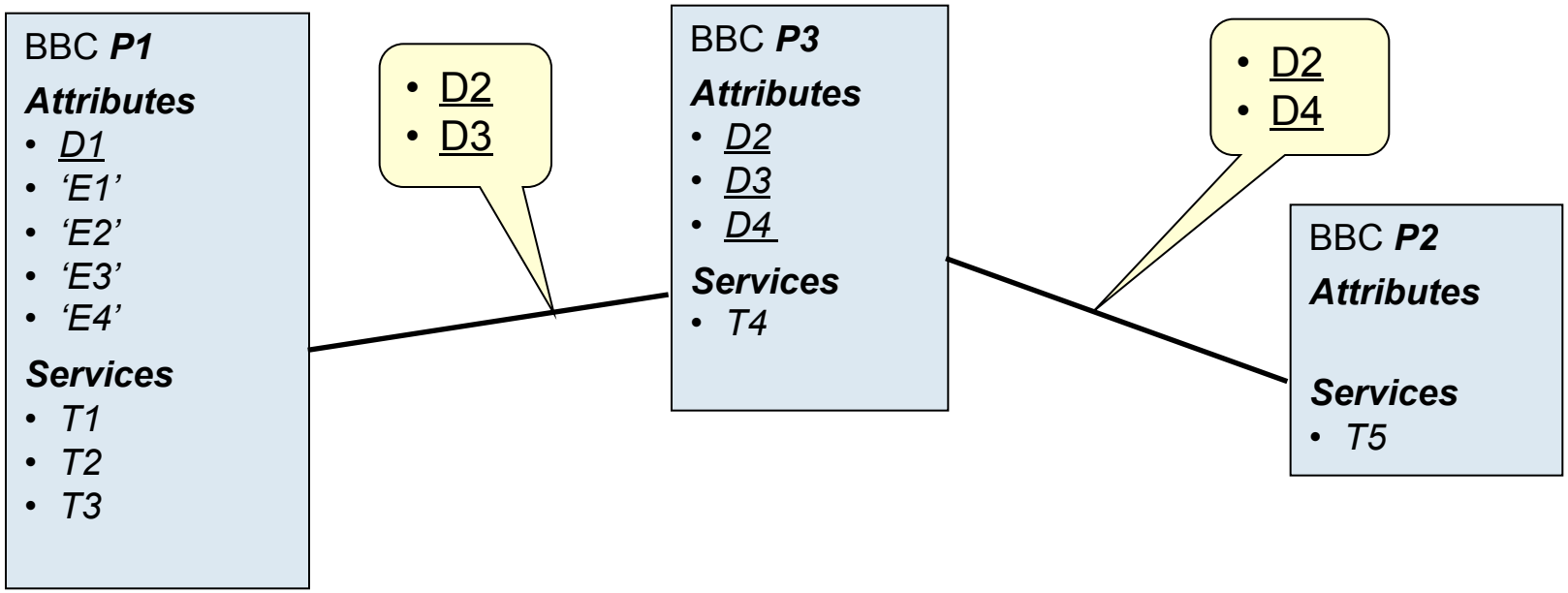


Strategy selected to correct Duplic e Attribute Error

- Assign data/event as Attribute to BBC that received responsibility for respective service (i.e., E3 and E4 used by T1 and T3, respectively)

In-Class EXAMPLE: Evaluating the BB after correcting the Missing Attribute Error

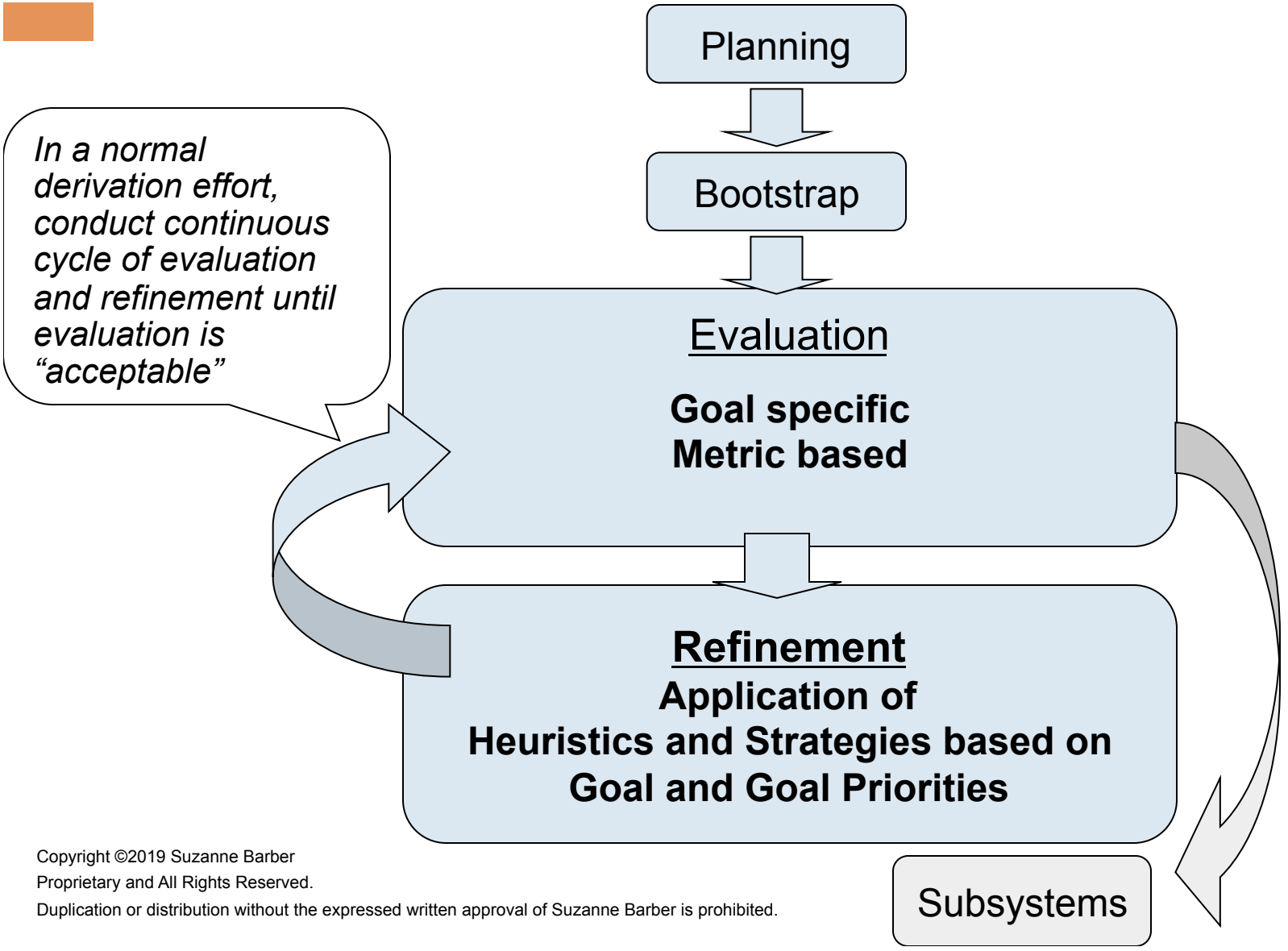
Graphic of BB with Missing Attribute Error corrected



Relevant Metrics indicating no Missing Attribute Error

- Num_Missing_Attributes(BB) = 0

Business Blueprint Derivation Process



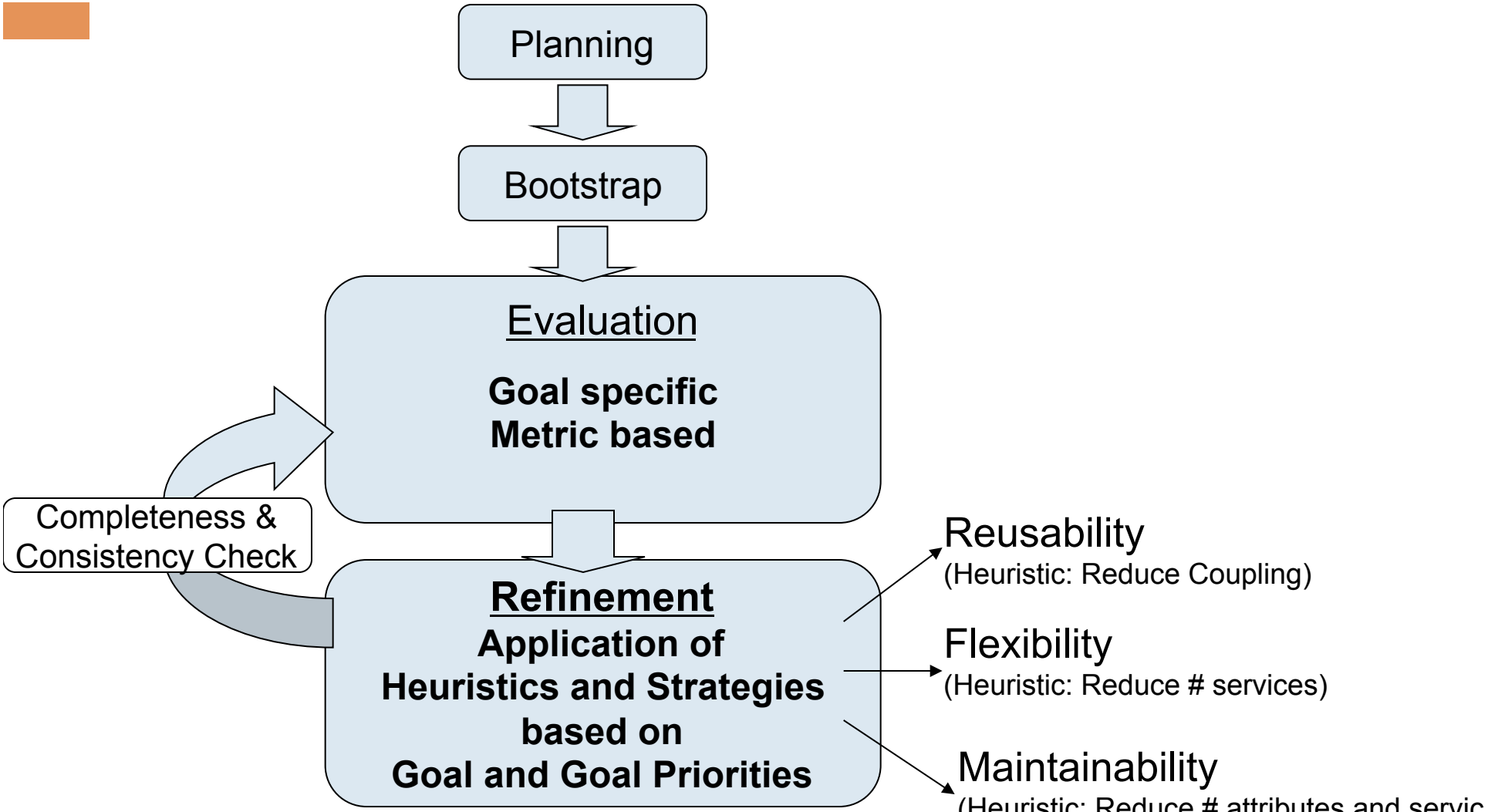
For example and discussion purposes

Starting with the Complete and Consistent BB derived from the Performer BB …

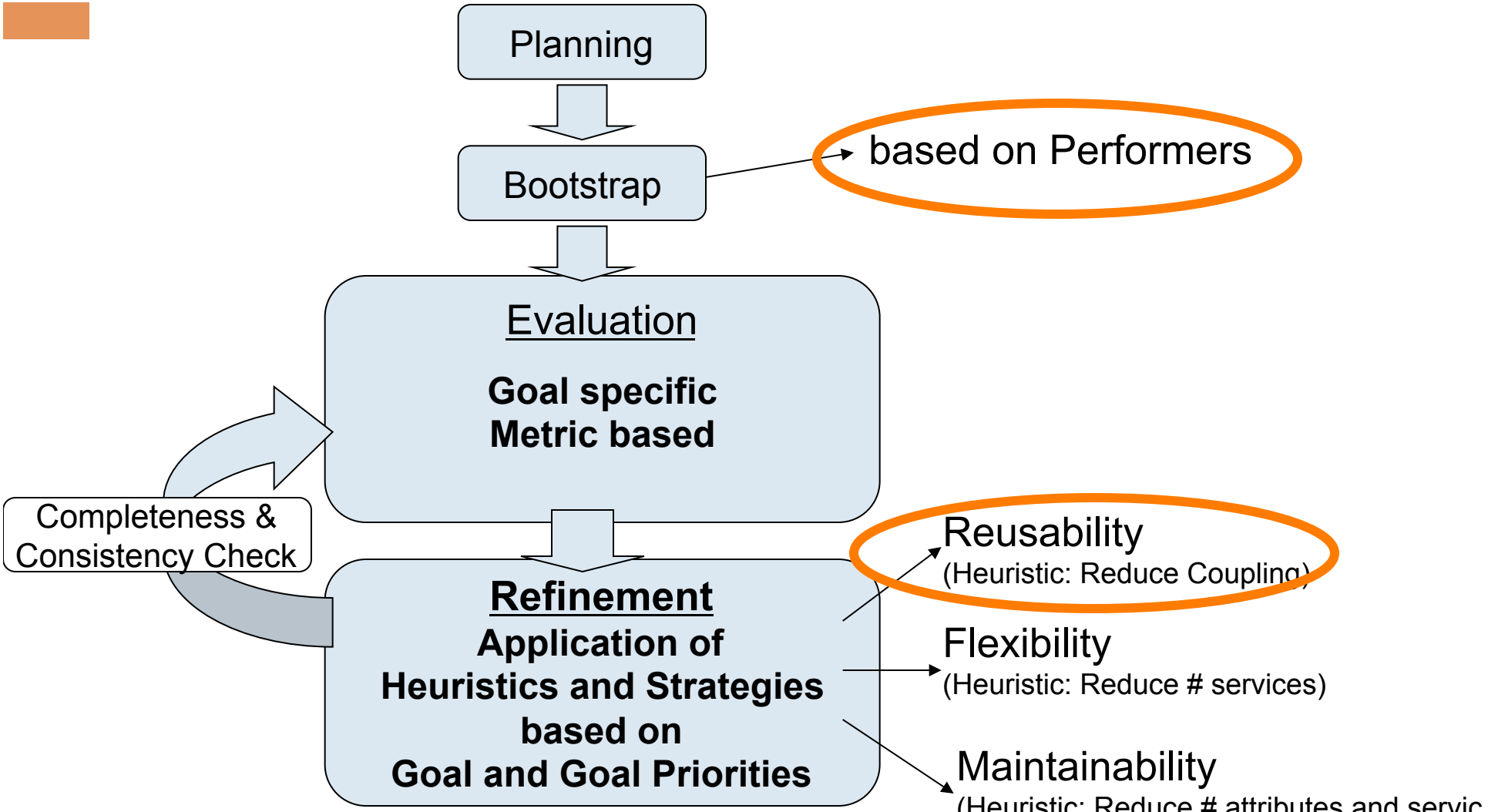
let's derive a "Reusable" BB

(essentially ignoring the Flexibility and Maintainability goals)

In-Class EXAMPLE: BB Derivation Process



In-Class EXAMPLE: BB Derivation Process

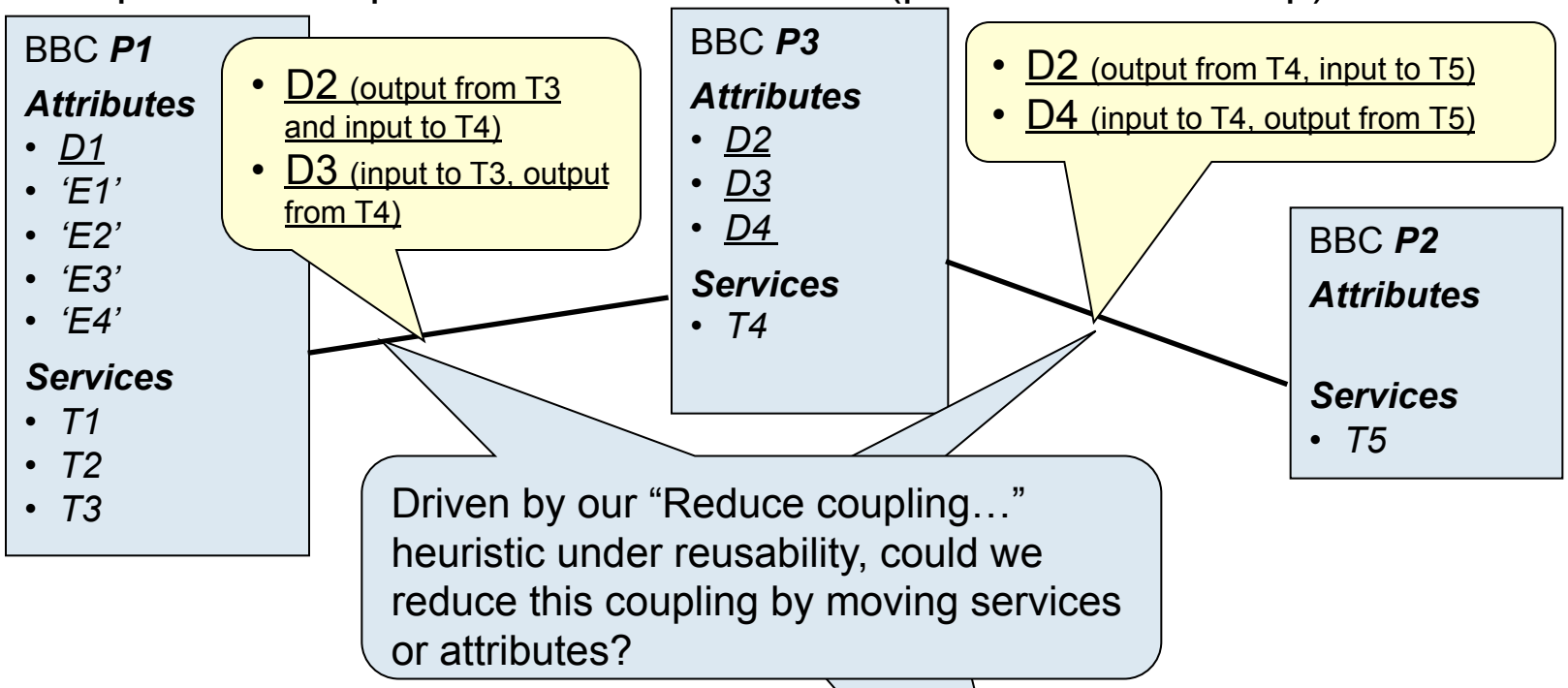


In-Class EXAMPLE: "Reusability" Refinement

- Using the "complete and consistent" BB
  - Evaluate the BB w.r.t. "Coupling," which impacts Reusability
  - Refine the BB to improve "Coupling"
  - Re-evaluate the refined BB w.r.t. "Coupling"

In-Class EXAMPLE: Evaluating "Reusability" Detecting Unacceptable Coupling Error

Graphic of "complete and consistent" BB (performer bootstrap)

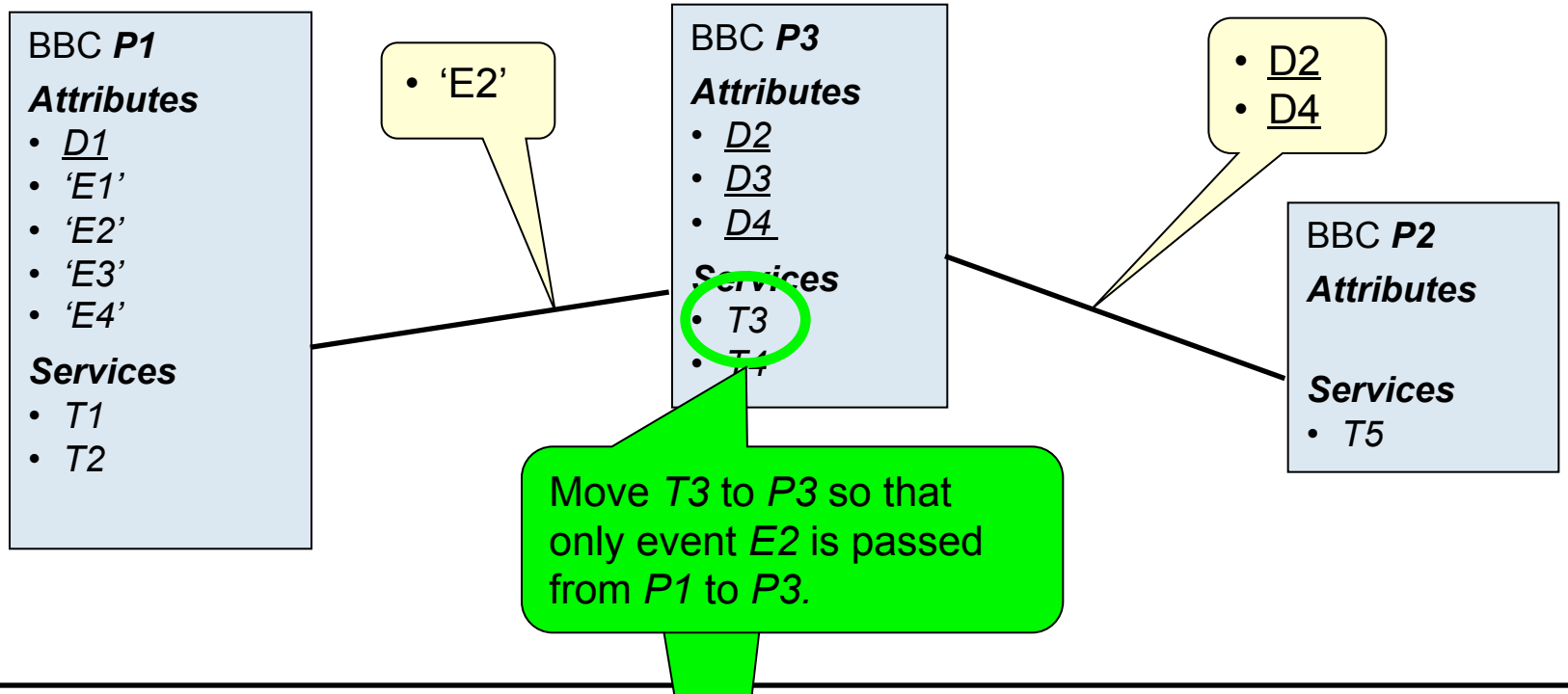


Relevant Metrics used to detect Unacceptable Coupling

- **Related to "Reduce coupling…" heuristic**
  - Num_Input/Output_between_BBCs(P1) = 2
  - Num_Input/Output_between_BBCs(P3) = 4
  - Num_Input/Output_between_BBCs(P2) = 2

In-Class EXAMPLE: Addressing the Unacceptable Coupling by moving services to BBCs most dependent on them

Graphic of BB with Unacceptable Coupling corrected

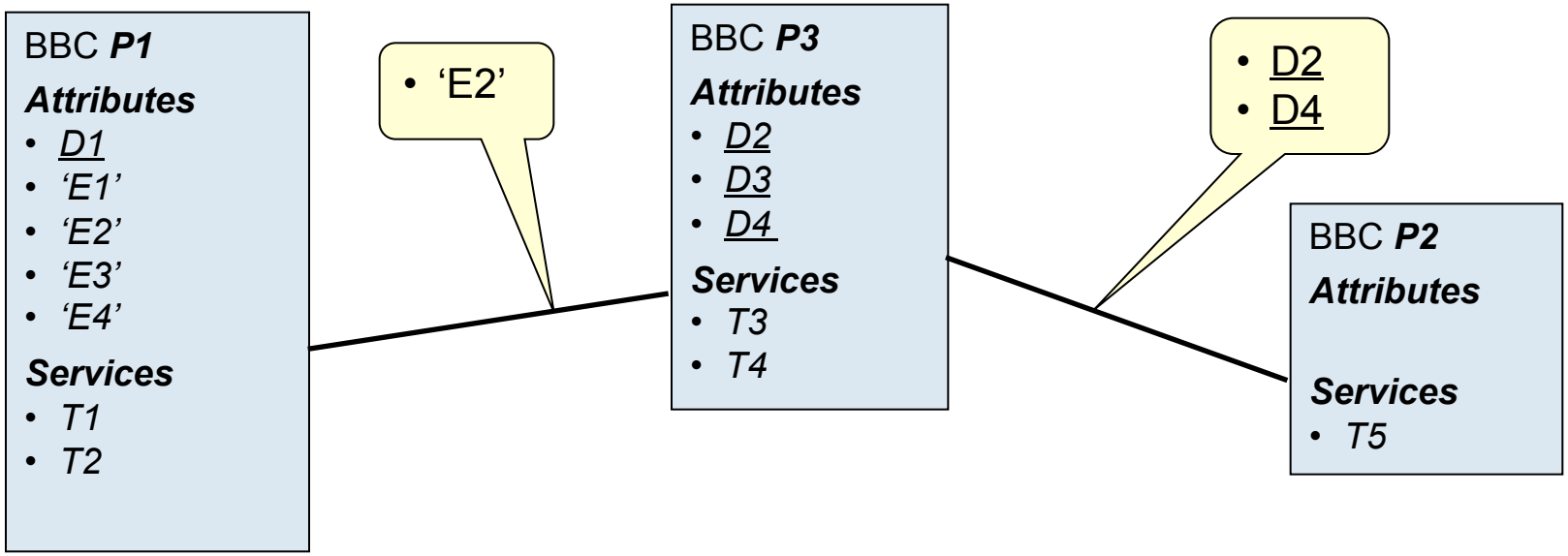


Strategy selected to address Unacceptable Coupling

- **Related to "Reduce coupling" heuristic:**
  - Move services to BBCs where they are used by the most other services (based on I/O)
    (i.e., move T3 to P3 to reduce I/O between P1 and P3)

In-Class EXAMPLE: Evaluating the Business Blueprint after reducing Coupling

Graphic of BB with Unacceptable Coupling corrected



Relevant Metrics indicating Reduced Coupling

- **Related to "Reduce coupling…" heuristic:**
  - Num_Input/Output_between_BBCs(P1) = 1
  - Num_Input/Output_between_BBCs(P3) = 3
  - Num_Input/Output_between_BBCs(P2) = 2

Per our heuristic, reduced coupling suggests better reusability.

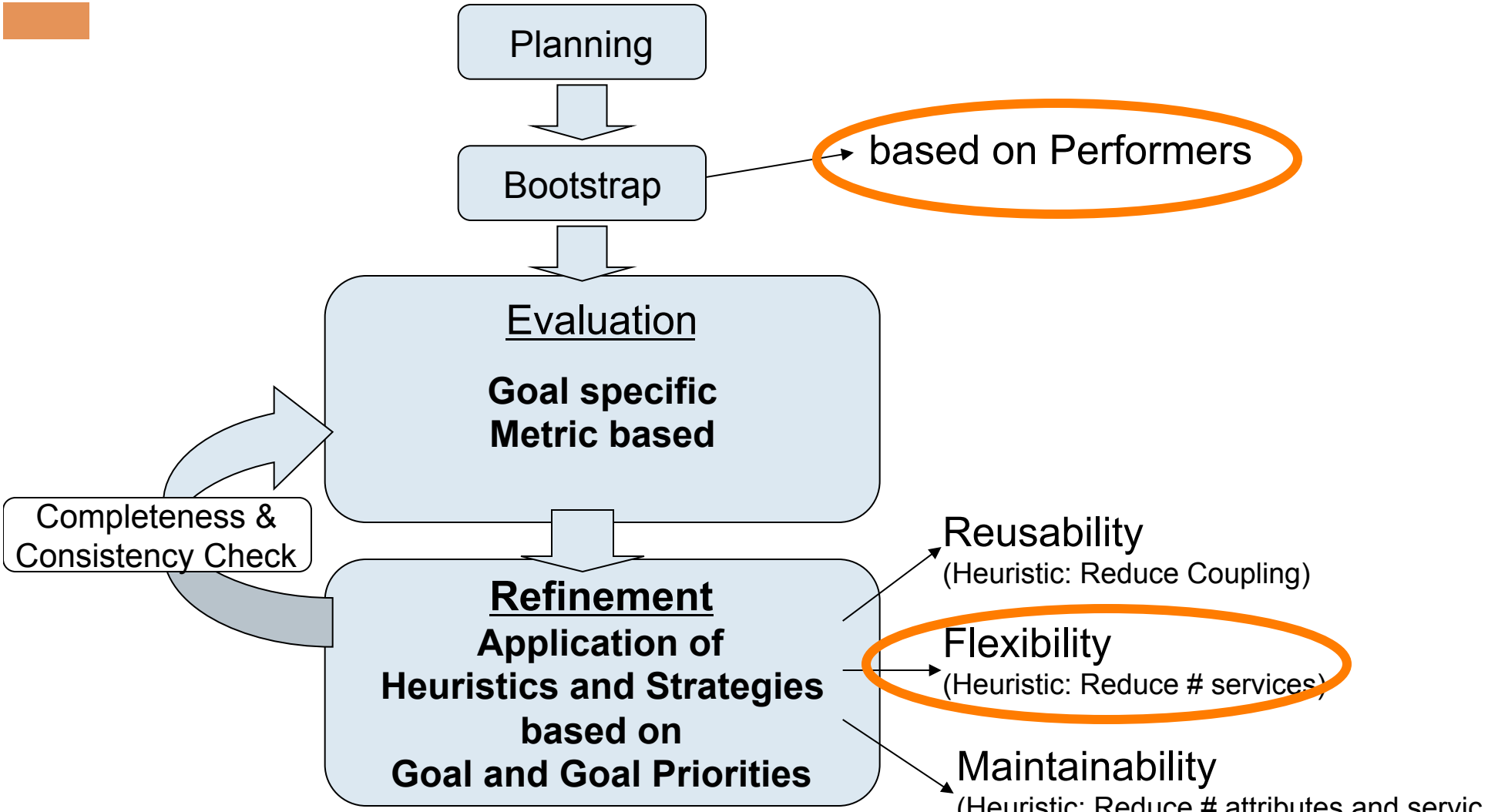
For example and discussion purposes

Starting with the Complete and Consistent BB derived from the Performer BB …

let's derive a "Flexible" BB

(essentially ignoring the Reusability and Maintainability goals)

In-Class EXAMPLE: Business Blueprint Derivation Process

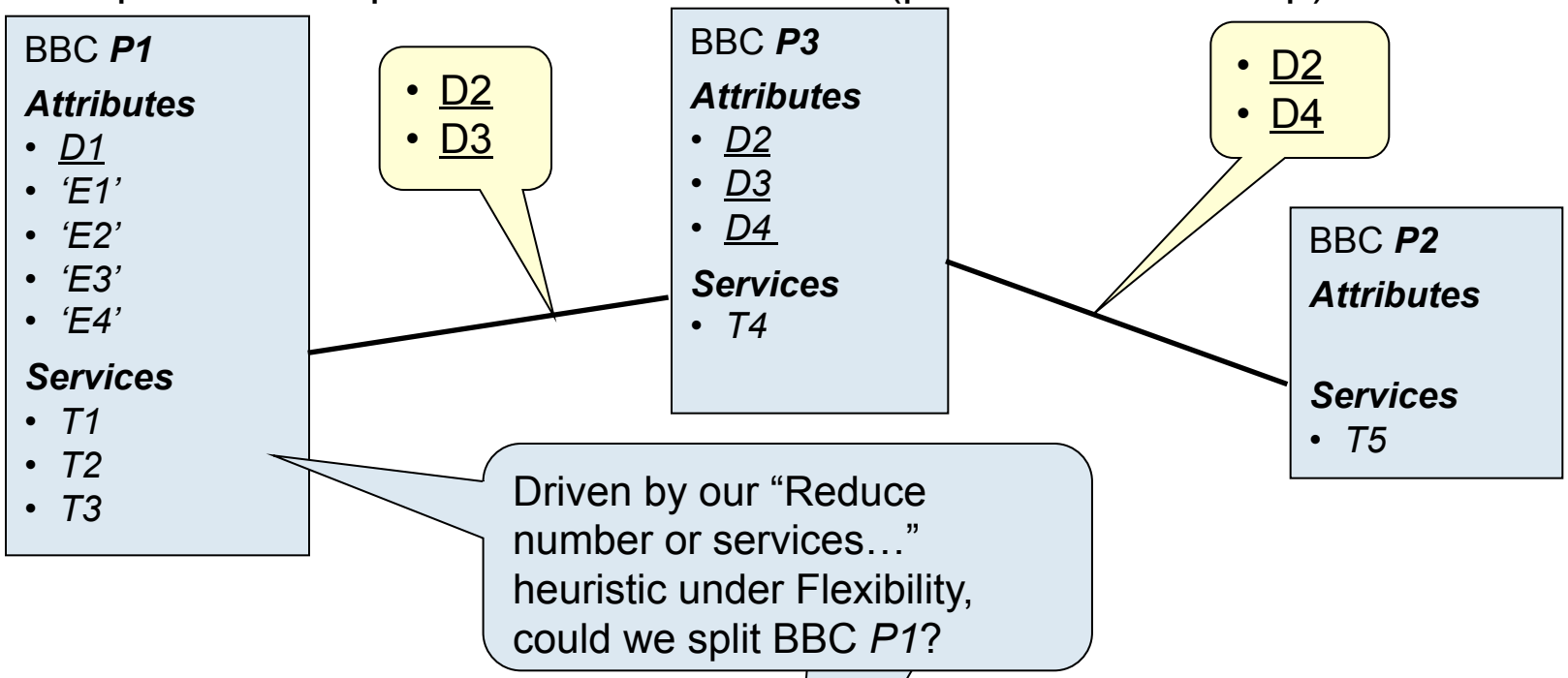


In-Class EXAMPLE: "Flexibility" Refinement

- Using the "complete and consistent" BB
  - Evaluate the BB w.r.t. "Num Services in BBC," which impacts Flexibility
  - Refine the BB to reduce "Num Services in BBC"
  - Re-evaluate the refined BB w.r.t. "Num Services in BBC"

In-Class EXAMPLE: Evaluating "Flexibility" Detecting Unacceptable Num Services in BBC

Graphic of "complete and consistent" BB (performer bootstrap)

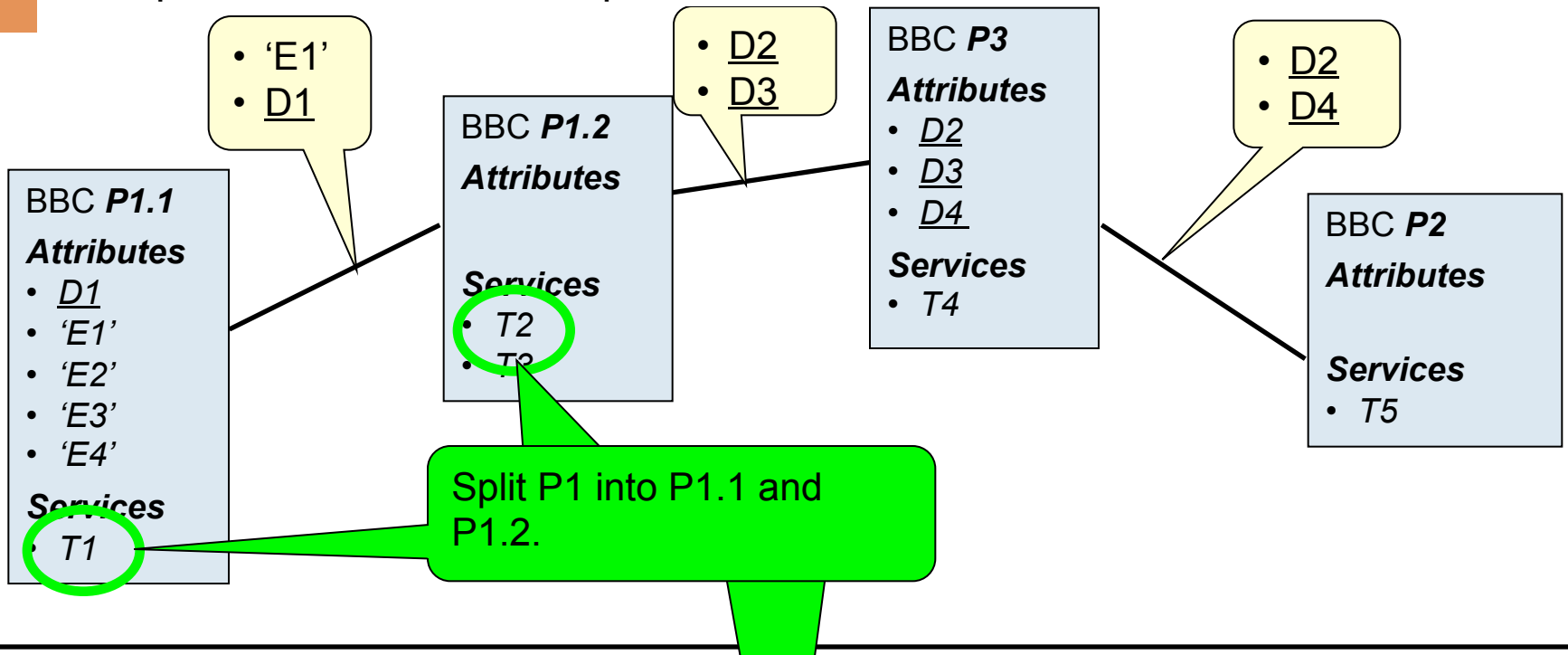


Relevant Metrics used to detect Unacceptable Num Services in BBC

- **Related to "Reduce number of services…" heuristic:**
  - Num_services_in_BBC(P1) = 3
  - Num_services_in_BBC(P2) = 1
  - Num_services_in_BBC(P3) = 1

In-Class EXAMPLE: Addressing the Unacceptable Num Services by splitting BBCs

Graphic of BB with Unacceptable Num Services in BBC corrected

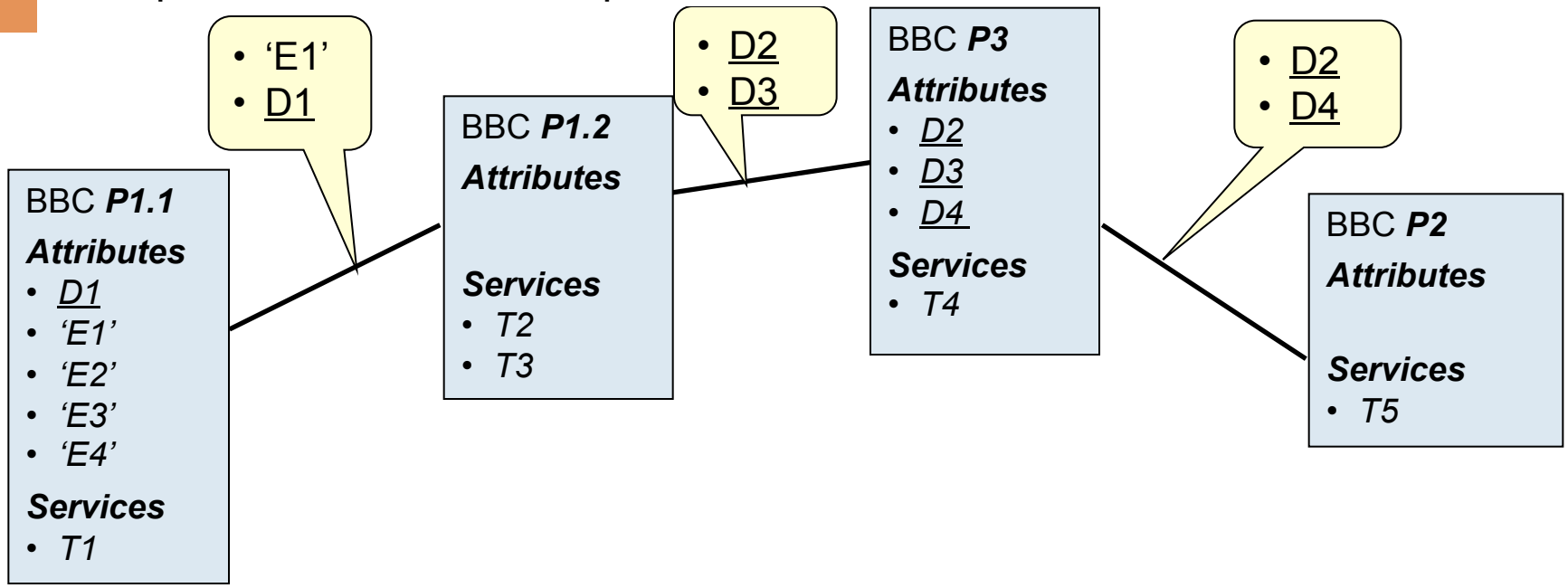


Strategy selected to address Unacceptable Num Services in BBC

- **Related to "Reduce number of services…" heuristic:**
  - Split BBCs to reduce size
    - (Split P1 into P1.1 and P1.2, with 1 and 2 tasks, respectively)

In-Class EXAMPLE: Evaluating the Business Blueprint after reducing Num Services in BBC

Graphic of BB with Unacceptable Num Services in BBC corrected

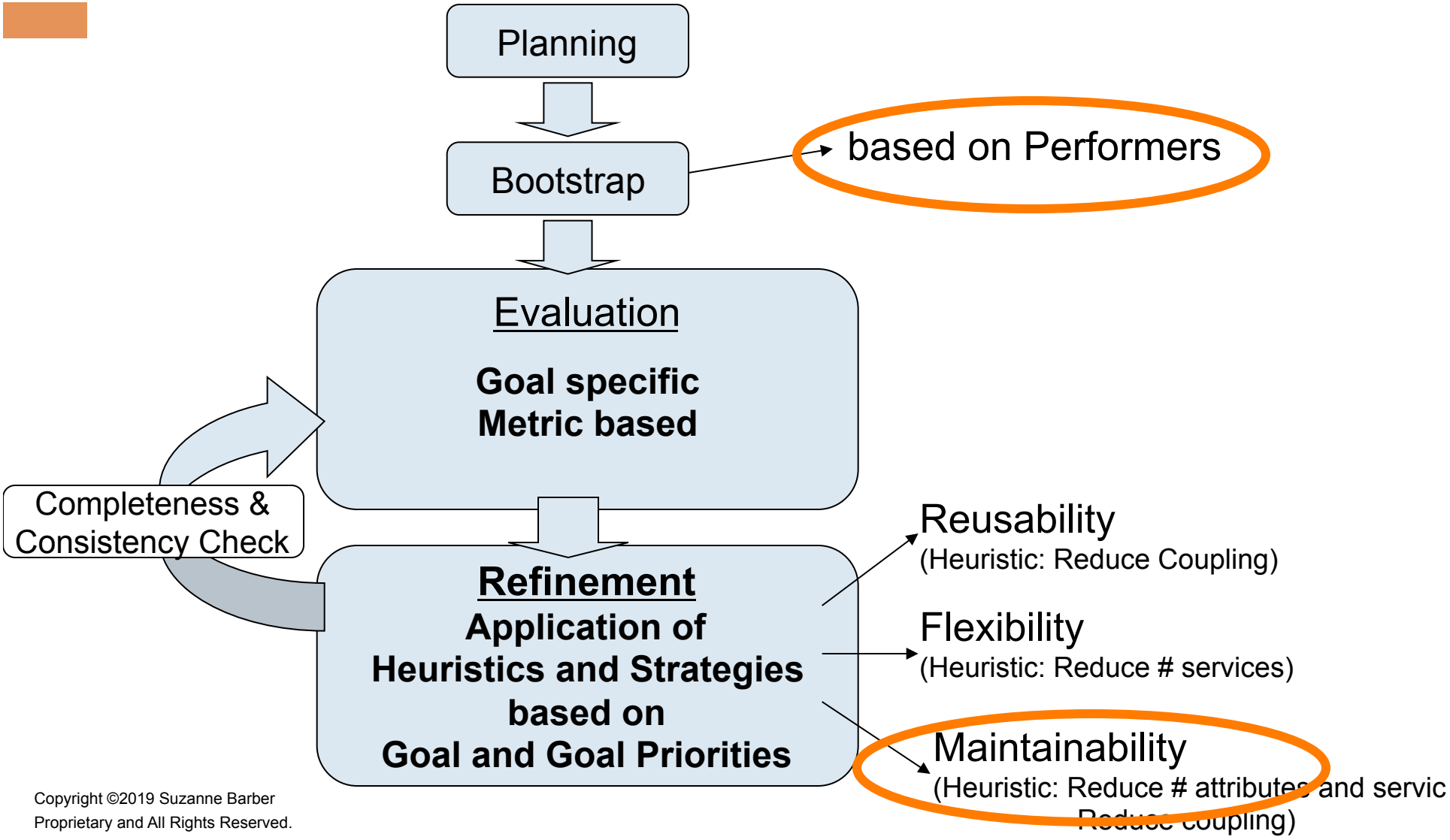


Relevant Metrics indicating reduced Number Services in BBC

- **Related to "Reduce number of services…" heuristic:**
  - Num_services_in_BBC(P1.1) = 1
  - Num_services_in_BBC(P1.2) = 2
  - Num_services_in_BBC(P3) = 1
  - Num_services_in_BBC(P2) = 1

Per our heuristic, reduced BBC size increases implementation flexibility.

In-Class EXAMPLE: BB Derivation Process

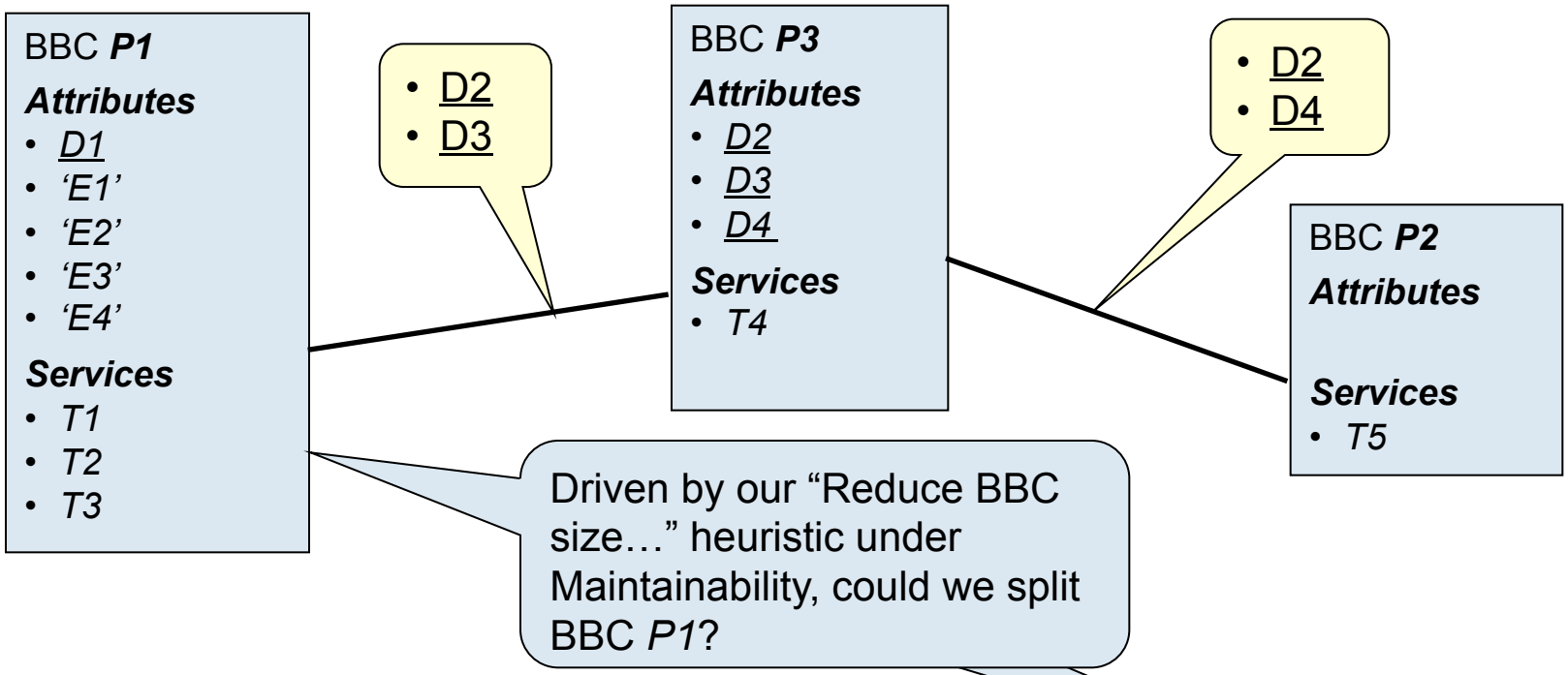


In-Class EXAMPLE:"Maintainability" Refinement

- Using the "complete and consistent" BB:
  - Evaluate the BB w.r.t. "Num Services in BBC," which impacts Maintainability
  - Refine the BB to reduce "Num Services in BBC"
  - Re-evaluate the refined BB w.r.t. "Num Services in BBC"
  - Note how reducing "Num Services in BBC" may increase "Coupling," which negatively impacts Maintainability
  - Evaluate the BB w.r.t. "Num Attributes in BBC," which impacts Maintainability
  - Refine the BB to reduce "Num Attributes in BBC"
  - Re-evaluate the refined BB w.r.t. "Num Attributes in BBC"

In-Class EXAMPLE: Evaluating "Maintainability" Detecting Unacceptable Num Services in BBC

Graphic of "complete and consistent" BB (performer bootstrap)

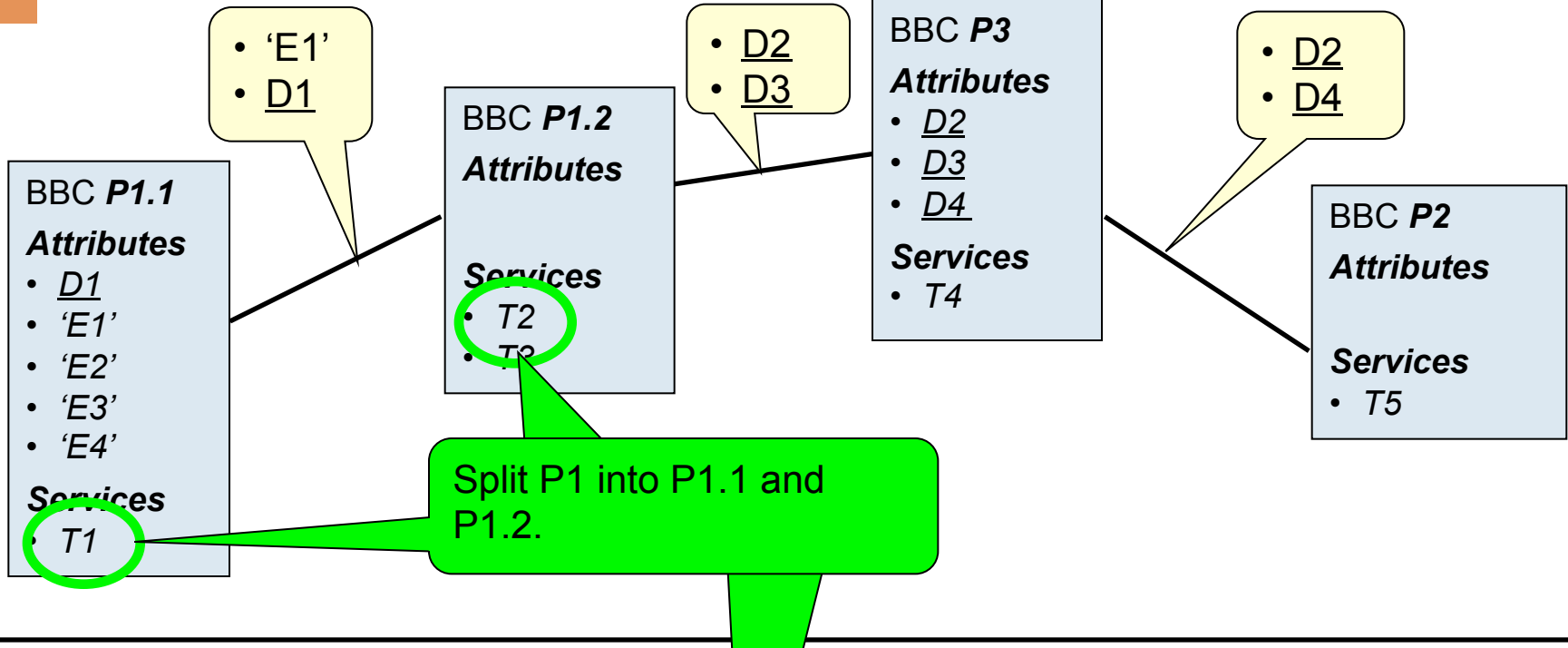


Maintainability Metrics (including those to detect Unacceptable Num Svcs)

- **Related to "Reduce coupling…" heuristic:**
  - **Num_Input/Output_between_BBCs(P1) = 2**
  - **Num_Input/Output_between_BBCs(P3) = 4**
  - **Num_Input/Output_between_BBCs(P2) = 2**
- **Related to "Reduce number of attributes…" heuristic:**
  - **Num_attributes_in_BBC(P1) = 5**
  - **Num_attributes_in_BBC(P3) = 3**
  - **Num_attributes_in_BBC(P2) = 0**
- **Related to "Reduce number of services…" heuristic:**
  - **Num_services_in_BBC(P1) = 3**
  - **Num_services_in_BBC(P3) = 1**
  - **Num_services_in_BBC(P2) = 1**

In-Class EXAMPLE: Addressing the Unacceptable Num Services by splitting BBCs

Graphic of BB with Unacceptable Num Services in BBC corrected

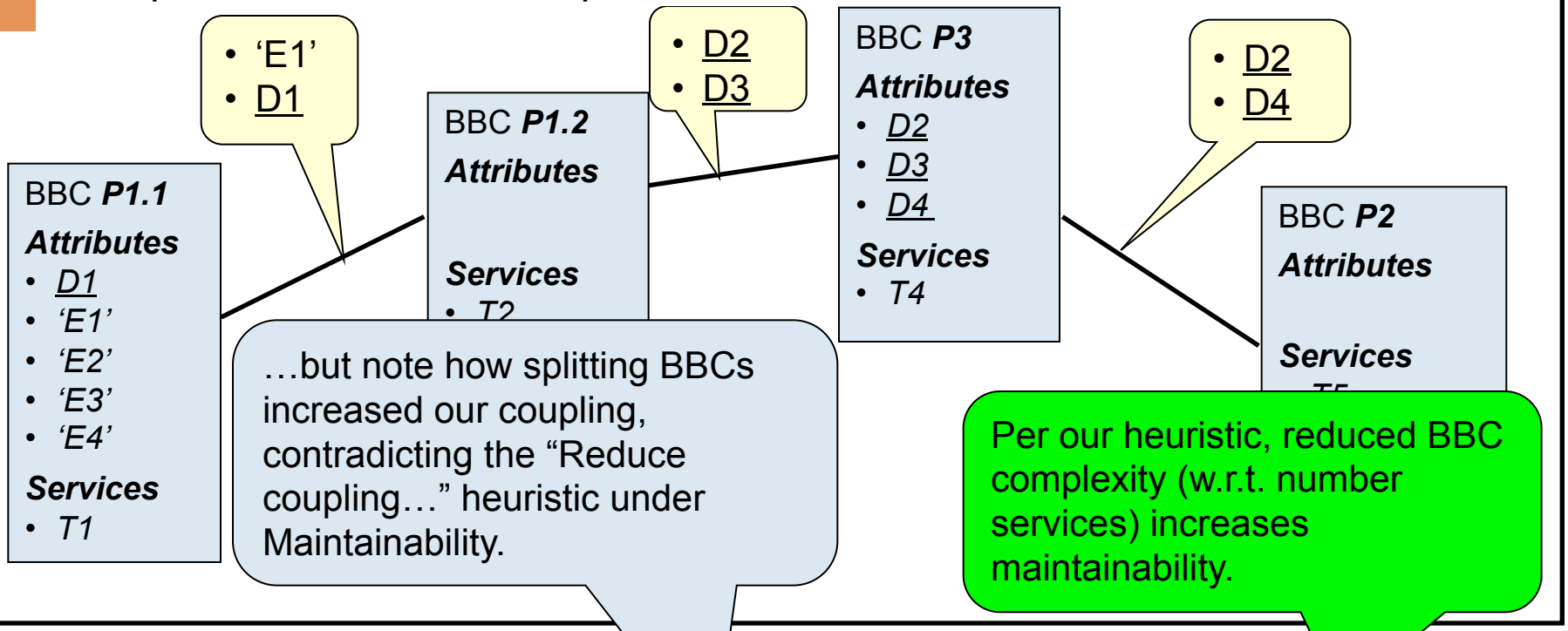


Strategy selected to address Unacceptable Num Services in BBC

- **Related to "Reduce number of Services…" heuristic:**
  - Split BBCs to reduce size
    - (Split P1 into P1.1 and P1.2, with 1 and 2 tasks, respectively)

In-Class EXAMPLE: Evaluating the Business Blueprint after reducing Num Services in BBC

Graphic of BB with Unacceptable Num Services in BBC corrected

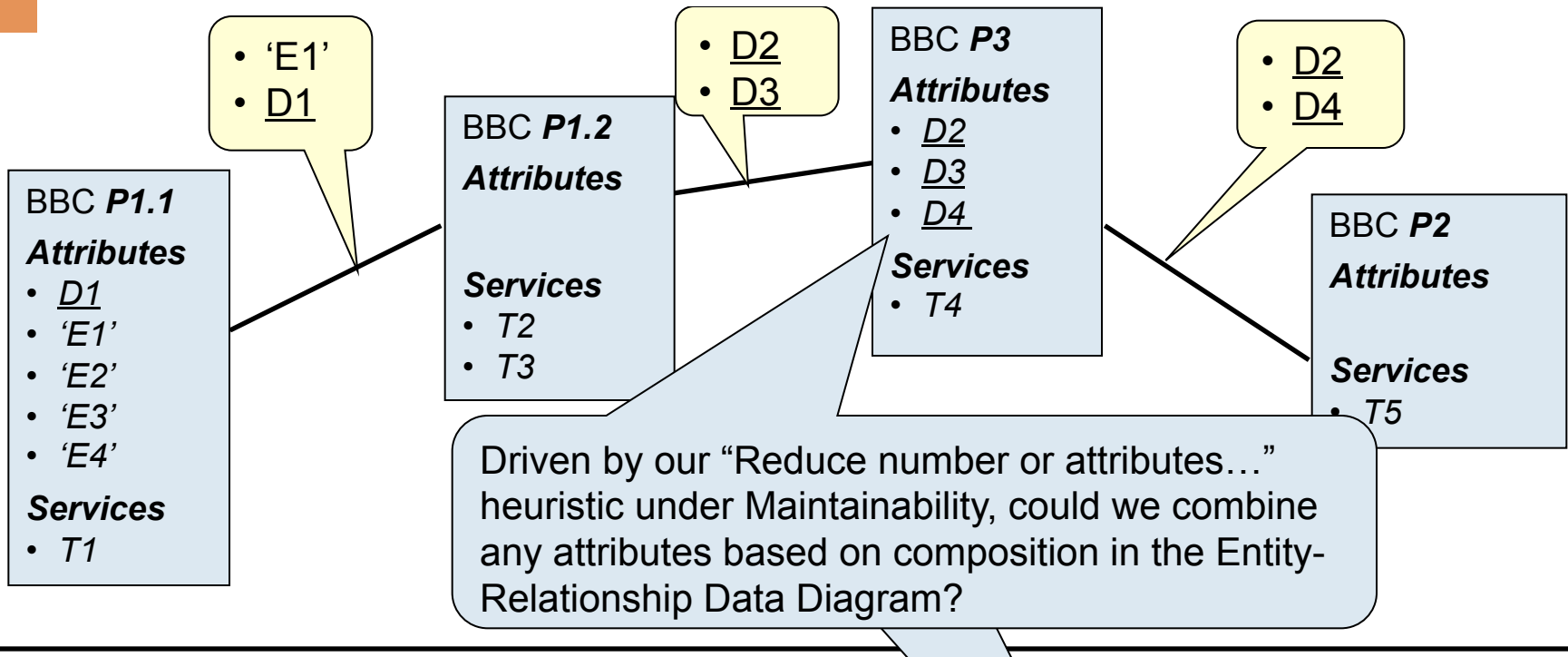


Relevant Metrics indicating red d Num Services yet *INCREASED* oupling

- **Related to "Reduce coupling…" heuristic:**
  - **Num_Input/Output_between_BBCs(P1.1) =**
  - **Num_Input/Output_between_BBCs(P1.2) = 4**
  - **Num_Input/Output_between_BBCs(P3) =**
  - **Num_Input/Output_between_BBCs(P2) = 2**
- **Related to "Reduce number of attributes…" heuristic:**
  - **Num_attributes_in_BBC(P1.1) = 5**
  - **Num_attributes_in_BBC(P1.2) = 0**
  - **Num_attributes_in_BBC(P3) = 3**
  - **Num_attributes_in_BBC(P2) = 0**
- **Related to "Reduce number of servi es…" heuristic:**
  - **Num_services_in_BBC(P1.1) = 1**
  - **Num_services_in_BBC(P1.2) = 2**
  - **Num_services_in_BBC(P3) = 1**
  - **Num_services_in_BBC(P2) = 1**

In-Class EXAMPLE: Evaluating "Maintainability" Detecting Unacceptable Num Attributes in BBC

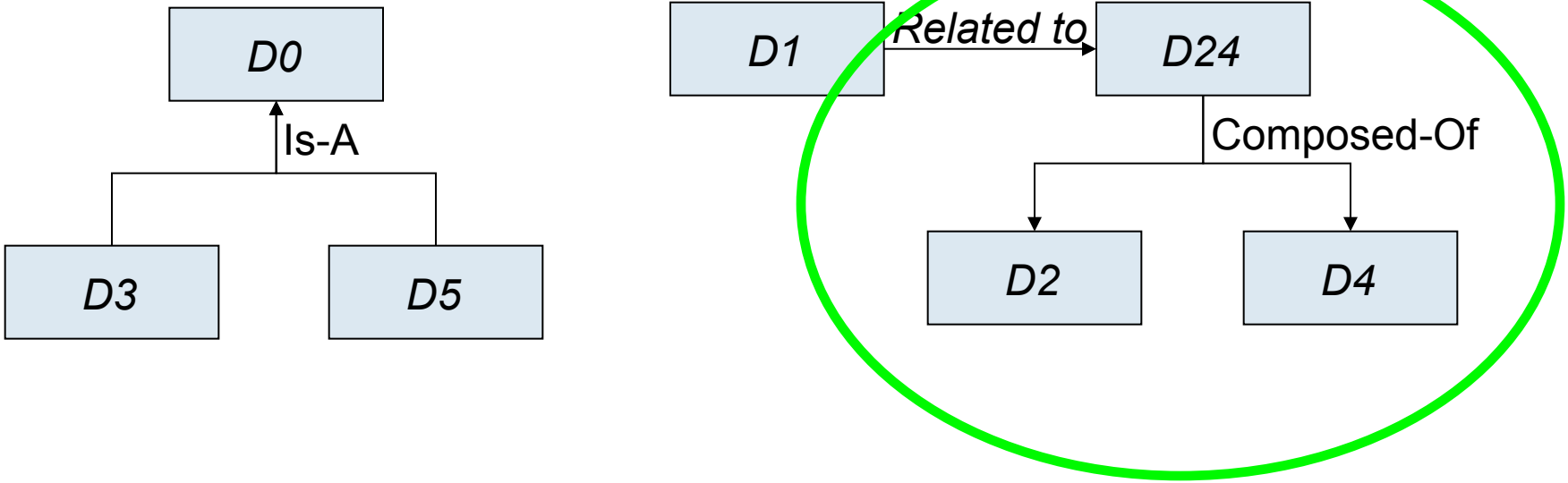
Graphic of refined "Maintainability" BB with services reduced



Maintainability Metrics (including those to de[illegible] Unacceptable Num Attrs)

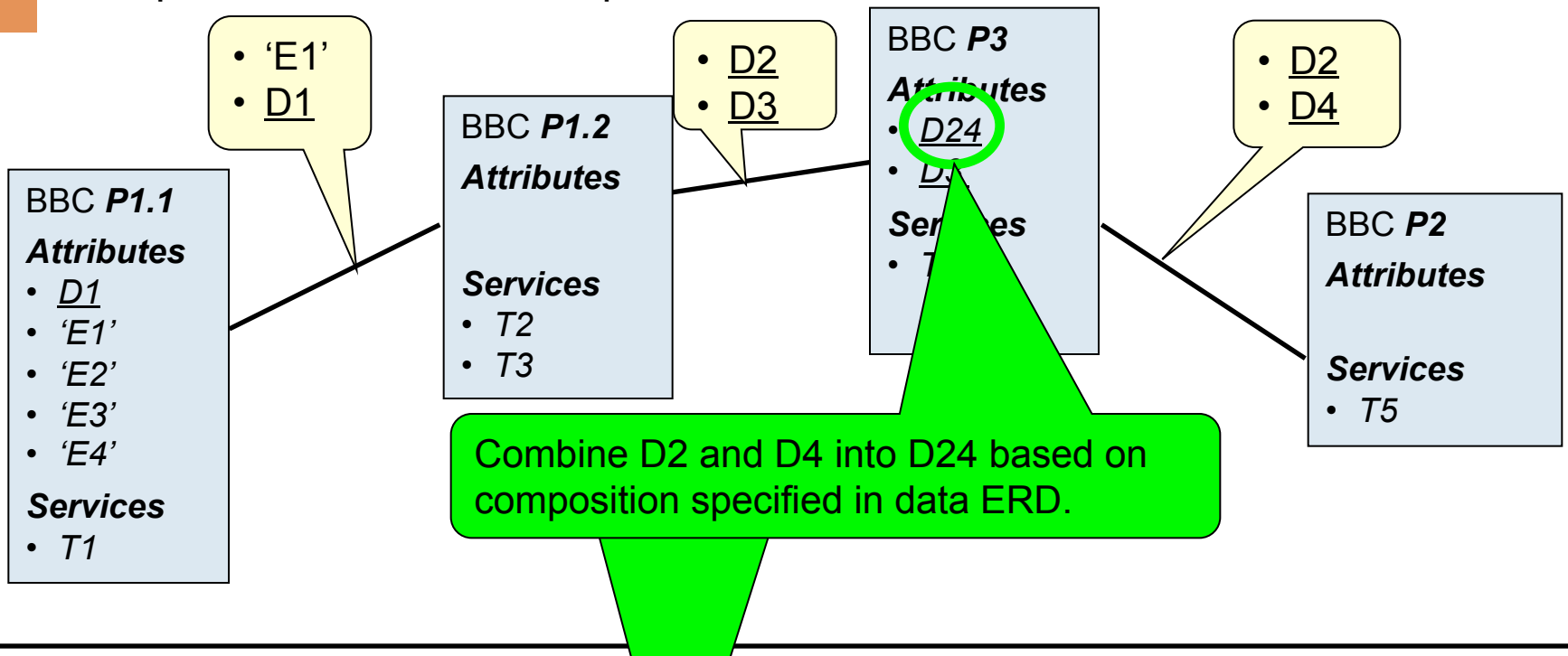
- **Related to "Reduce coupling…" heuristic:**
  - **Num_Input/Output_between_BBCs(P1.1) = 2**
  - **Num_Input/Output_between_BBCs(P1.2) = 4**
  - **Num_Input/Output_between_BBCs(P3) = 4**
  - **Num_Input/Output_between_BBCs(P2) = 2**
- **Related to "Reduce number of attributes…" heuristic:**
  - **Num_attributes_in_BBC(P1.1) = 5**
  - **Num_attributes_in_BBC(P1.2) = 0**
  - **Num_attributes_in_BBC(P3) = 3**
  - **Num_attributes_in_BBC(P2) = 0**
- **Related t[illegible]educe number of services…" heuristic:**
  - **Num[illegible]vices_in_BBC(P1.1) = 1**
  - **Num_s[illegible]ices_in_BBC(P1.2) = 2**
  - **Num_ser[illegible]ces_in_BBC(P3) = 1**
  - **Num_servi[illegible]s_in_BBC(P2) = 1**

In-Class EXAMPLE: Addressing the Unacceptable Num Attributes examining the Data ERD for possible composition



In-Class EXAMPLE: Addressing the Unacceptable Num Attributes by combining attributes based on composition

Graphic of BB with Unacceptable Num Attributes in BBC corrected

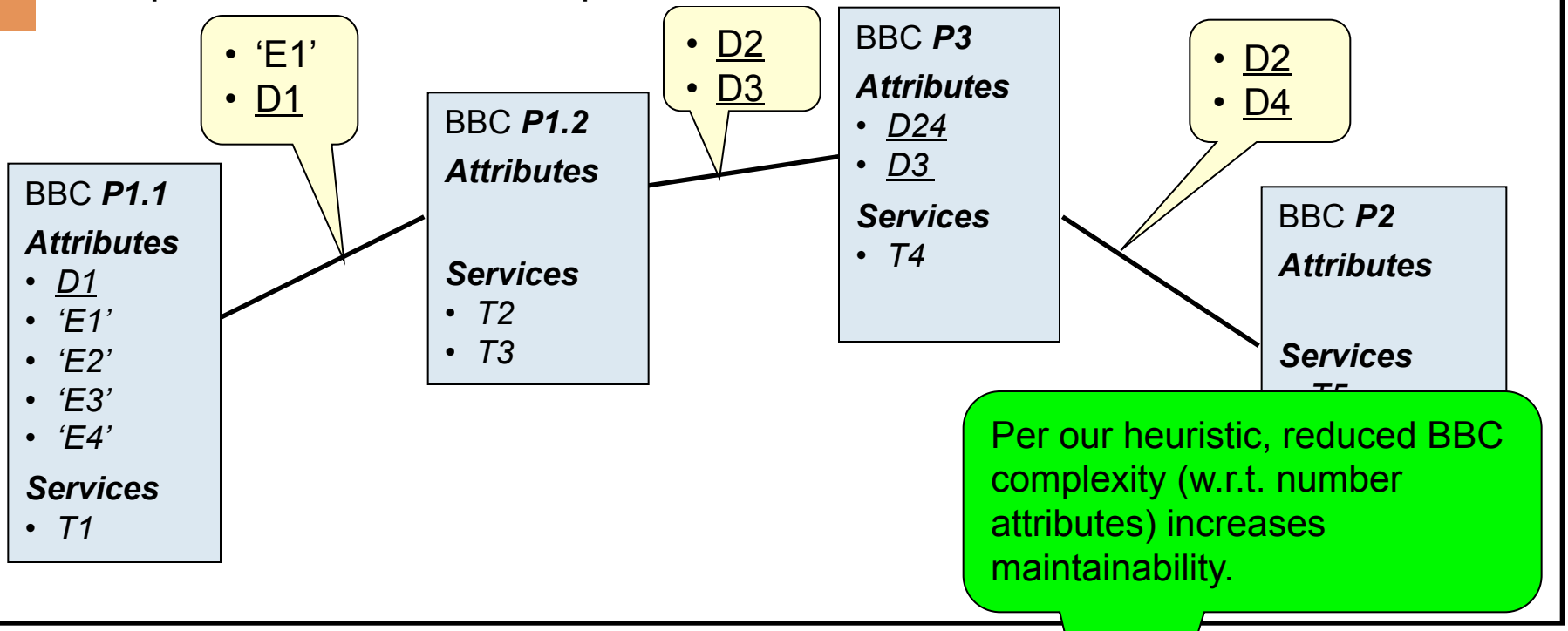


Strategy selected to address Unacceptable Num Attributes in BBC

- **Related to "Reduce number of attributes…" heuristic:**
  - **Group data attributes based on composition (D24 is composed of D2 and D4)**

In-Class EXAMPLE: Evaluating the BB after reducing Num Attributes in BBC

Graphic of BB with Unacceptable Num Attributes in BBC corrected



Relevant Metrics indicating reduced Num Attributes in

- **Related to "Reduce coupling…" heuristic:**
  - **Num_Input/Output_between_BBCs(P1.1) = 2**
  - **Num_Input/Output_between_BBCs(P1.2) = 4**
  - **Num_Input/Output_between_BBCs(P3) = 4**
  - **Num_Input/Output_between_BBCs(P2) = 2**
- **Related to "Reduce number of attributes…" heuristic:**
  - **Num_attributes_in_BBC(P1.1) = 5**
  - **Num_attributes_in_BBC(P1.2) = 0**
  - **Num_attributes_in_BBC(P3) = 2**
  - **Num_attributes_in_BBC(P2) = 0**
- **Related to "Reduce number of services…" heuristic:**
  - **Num_services_in_BBC(P1.1) = 1**
  - **Num_services_in_BBC(P1.2) = 2**
  - **Num_services_in_BBC(P3) = 1**
  - **Num_services_in_BBC(P2) = 1**

For example and discussion purposes

Starting with the Complete and Consistent BB derived from the Performer BB …

let's derive a "Flexible" BB

(essentially ignoring the Reusability and Maintainability goals)

In-Class EXAMPLE: Adding a subsystem to the Business Blueprint refined w.r.t. "Maintainability"

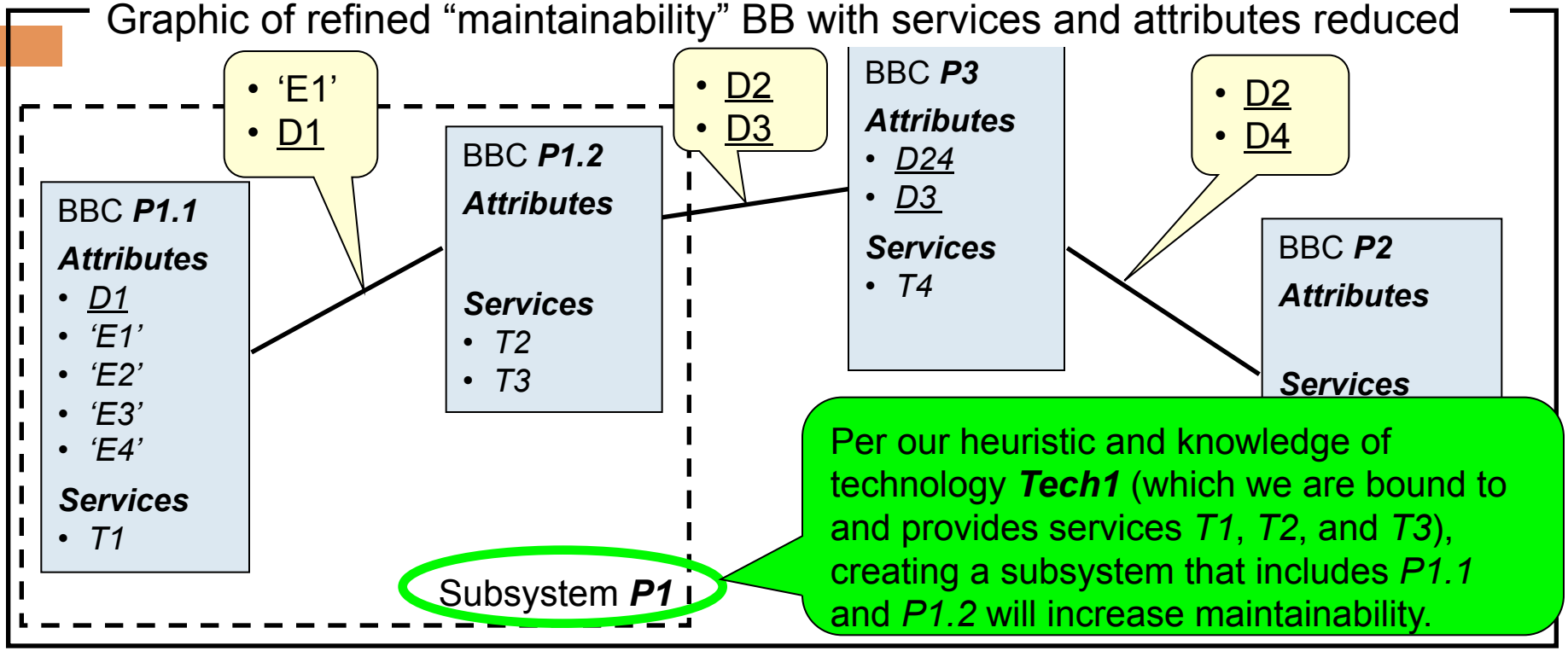
- Subsystems allow you to define collections of Business Blueprint Components (BBC) that contribute to your goals
- Often, when goal conflicts arise, subsystems can help achieve lower priority goals and heuristics since the BBC allocation reflects higher priority goals and heuristics
- This example shows how a subsystem might be justified in the context of "Maintainability"

In-Class EXAMPLE: Extending the heuristics, strategies, and metrics under "Maintainability" to consider subsystems

- One approach for achieving "Maintainability" goal:
  - Heuristic: "Reduce number of services per technology to simplify technology complexity"
    - Strategies for achieving heuristic:
      - Split BBCs to reduce size
  - Heuristic: "Reduce number of attributes per technology to simplify technology complexity"
    - Strategies for achieving heuristic:
      - Group data attributes based on composition
  - Heuristic: "Reduce coupling between technologies, thereby reducing interface complexity"
    - Strategies for achieving heuristic:
      - Move services to BBCs where to the BBCs where they are used by the most other services
      - Move attributes to BBCs where they are most used by services
  - Heuristic: "Align BBCs with existing, highly influential technologies, since the maintainability of the integrated system is highly dependent on the installation of those technologies"
    - Strategies for achieving heuristic:
      - For a given highly influential technology, define a subsystem that groups BBCs according to the services the technology can provide
- Means for evaluating BB w.r.t. "Maintainability"
  - Size/Complexity Metric
    - "Num Services in BBC"
      - For each BBC:
        - Number of services in BBC
    - "Num Attributes in BBC"
      - For each BBC:
        - Number of attributes in BBC
  - Coupling Metric
    - "Num Input/Output between BBCs"
      - For each BBC:
        - Num service inputs that come from another BBC + Num service outputs that go to another BBC
  - Subsystem Metric
    - "Degree to which subsystems reflect influential technologies"

In-Class EXAMPLE: Defining a subsystem to improve "Maintainability"

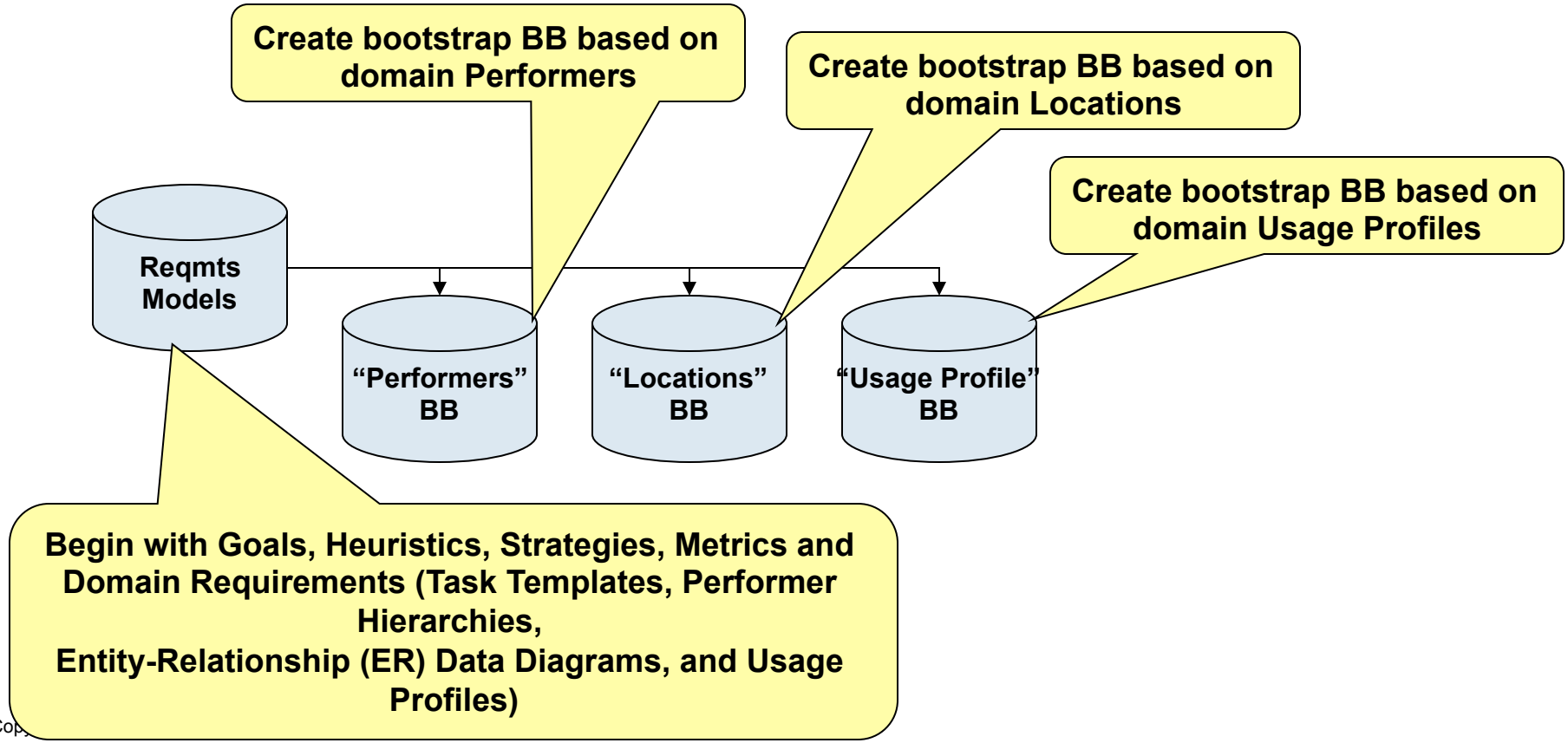
Graphic of refined "maintainability" BB with services and attributes reduced



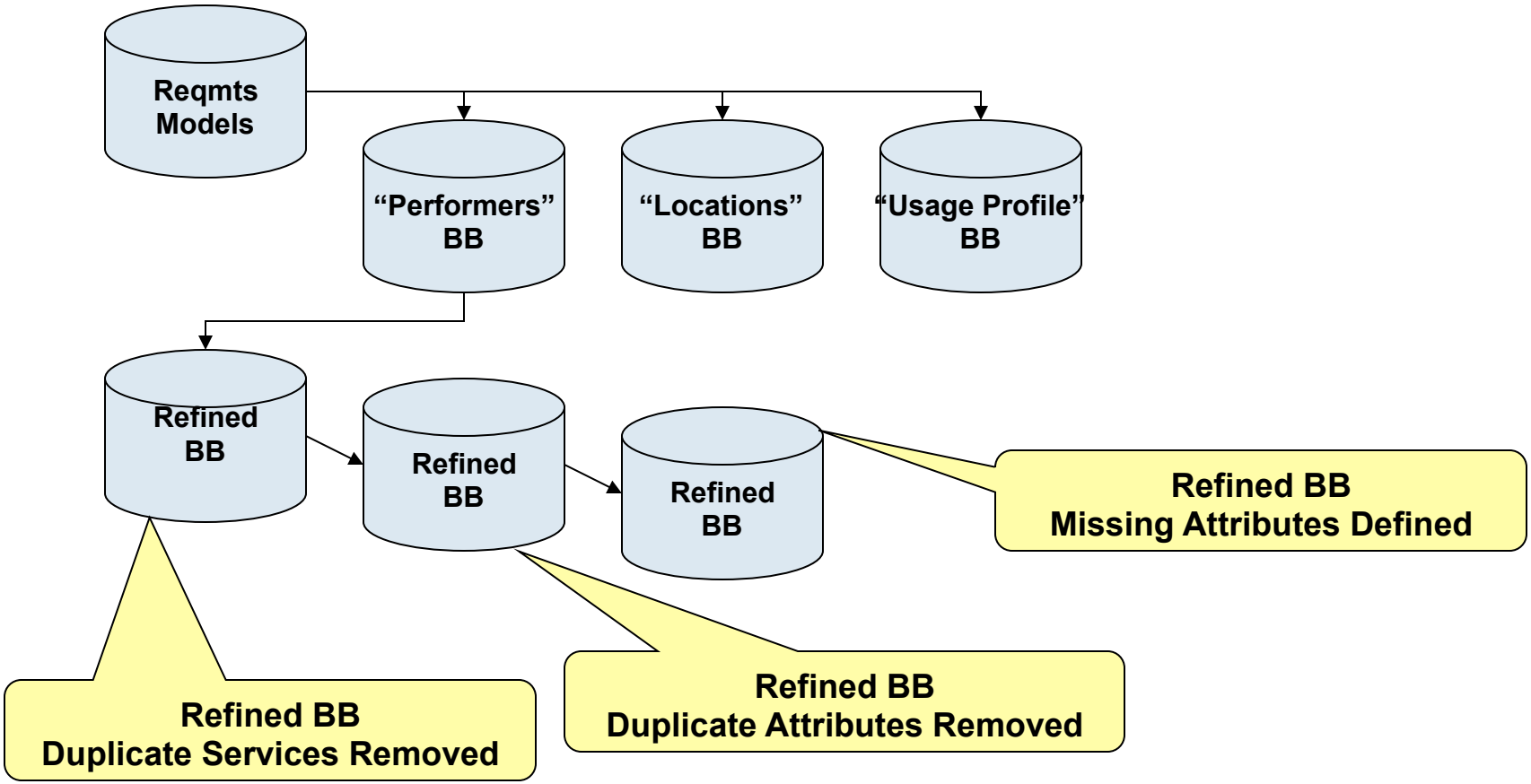
Maintainability Metrics

- **Related to "Reduce coupling…" heuristic:**
  - **Num_Input/Output_between_BBCs(P1.1) = 2**
  - **Num_Input/Output_between_BBCs(P1.2) = 4**
  - **Num_Input/Output_between_BBCs(P3) = 4**
  - **Num_Input/Output_between_BBCs(P2) = 2**
- **Related to "Reduce number of services…" heuristic:**
  - **Num_services_in_BBC(P1.1) = 1**
  - **Num_services_in_BBC(P1.2) = 2**
  - **Num_services_in_BBC(P3) = 1**
  - **Num_services_in_BBC(P2) = 1**
- **Related to "Reduce number of attributes…" heuristic:**
  - **Num_attributes_in_BBC(P1.1) = 5**
  - **Num_attributes_in_BBC(P1.2) = 0**
  - **Num_attributes_in_BBC(P3) = 2**
  - **Num_attributes_in_BBC(P2) = 0**

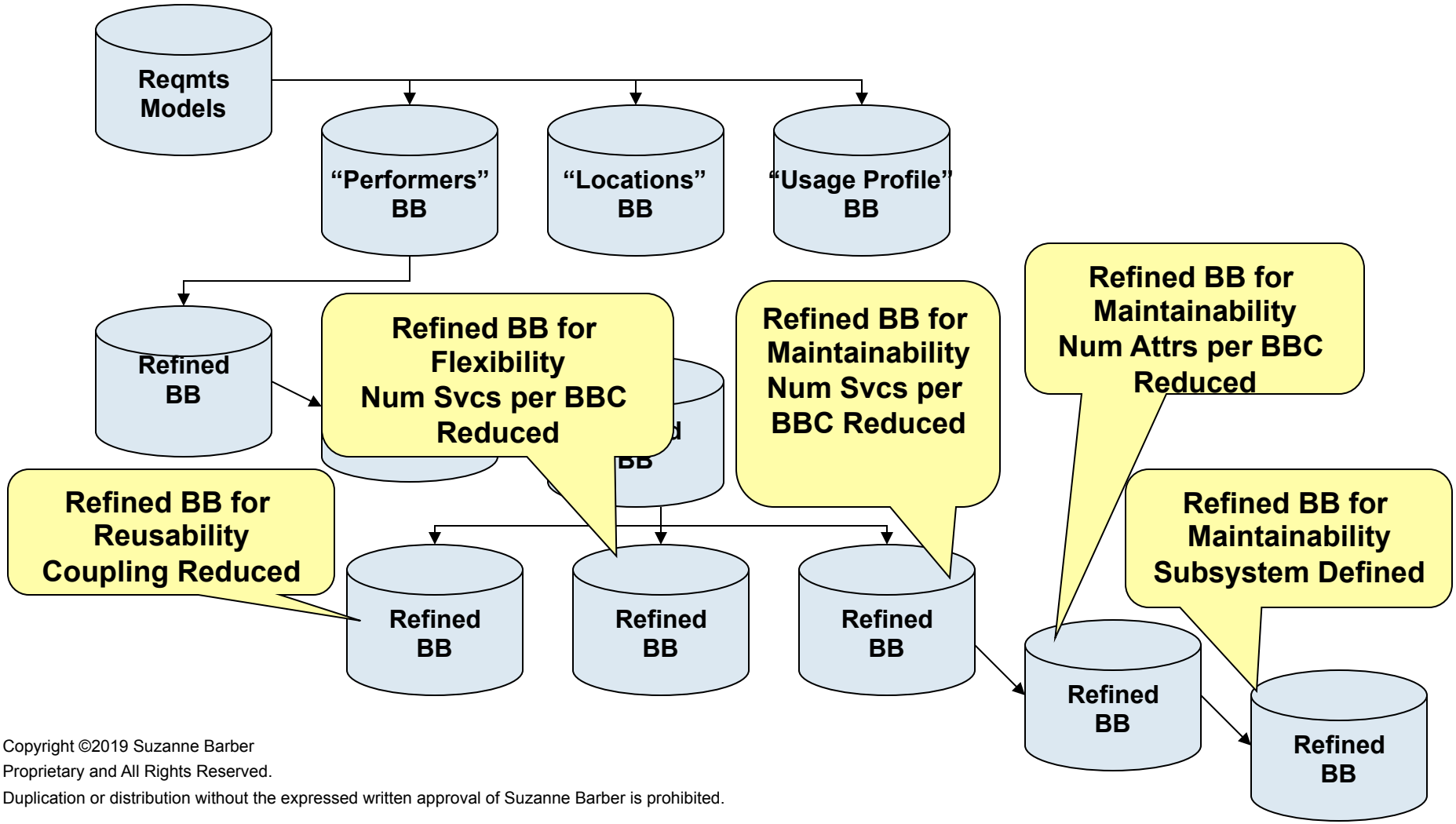
In-Class EXAMPLE: Traceability across BB Derivation Process



In-Class EXAMPLE: Traceability across BB Derivation Process



In-Class EXAMPLE: Traceability across Business Blueprint Derivation Process



Business Blueprint Components (BBCs)

- Comprise the Business Blueprint definition
- Component-based representation of functional, data, timing and dependency requirements
- Assigned one (or more) primitive responsibilities for
  - domain tasks (BBC services) and domain data and events (BBC attributes)
- Primitive responsibilities specify external view of BBC ("services offered" and "attributes owned")
- Constraints on primitive responsibilities specify
  - Service constraints: 1) Input and output data/events and 2) pre- and post- condition on services
  - Attribute constraints: 1) Value constraints and 2) Cardinality
- Building blocks for the Solution Blueprint where technology components become "instances" of the BBCs

Comments

- Do not make decisions based on assumptions. Only use…
  - Goals you know
  - Content you know
- Where does the rationale lie?
  - How do metrics fair w.r.t. goals (i.e., what metrics are notable)? Do they suggest a refinement (i.e., another iteration)?
    - For BB: Your overall approach for BB that will best achieve your goals and what you might be giving up (e.g., for initial derivation, "I think reusability will be best achieved by aligning BBCs according to performer roles. However, this may affect maintainability because…"). You should describe your approach terms of goals, heuristics, and strategies.
    - For each BBC: Why the BBC should have these attributes and services (e.g., in a subsequent refinement, "I will move Svc1 from BBCx to BBCy to reduce coupling…")
- While this presentation focused on the BB, the same notions of a step-by-step process and focused evaluation apply to all architectures.

Answers to Questions about Assignment #2

- Degree of Cohesion measurement

  (#Inputs received from functions within component or attributes within component + # outputs sent to functions within component or attributes within component)

  (ALL inputs for ALL functions within the component + ALL outputs for ALL functions within the component)

- Do we need to prioritize all the qualities from Assignment 1 or just the important qualities for this assignment ? Prioritize all qualities
- It is mentioned that we need to include Stakeholder needs in the table, do we need to describe the qualities for this column? Yes. In assignment #1, you may have specified the quality directly …. E.g. Reliability. In this case the quality is clear… Reliability. Or you may have said the quality/constraint is "System must not fail more than 1 hour per month" in this case, relate the need to a quality … "System must not fail more than 1 hour per month" = Reliability
- One of the important quality is Availability of the system, but I am not sure if any heuristic fits in this category, as I feel this is depends on deployment (like backup servers). How to include this quality as my goal, it will be great if you provide me some directions. You are correct that this quality is a challenge. It tends to be a dynamic NFR that is more difficult to address with blueprint structure. One idea: distribute frequently used functions across BBC. This separation will allow for distribution across servers in the Deployment Blueprint.
- Part 3
  - This part requires calculation of the structural metrics. Are we expected to write numbers as answers for this section ? Yes.