Architecture design process

from the book:

Designing software architectures by Cervantes and Kazman

Attribute driven design (ADD)

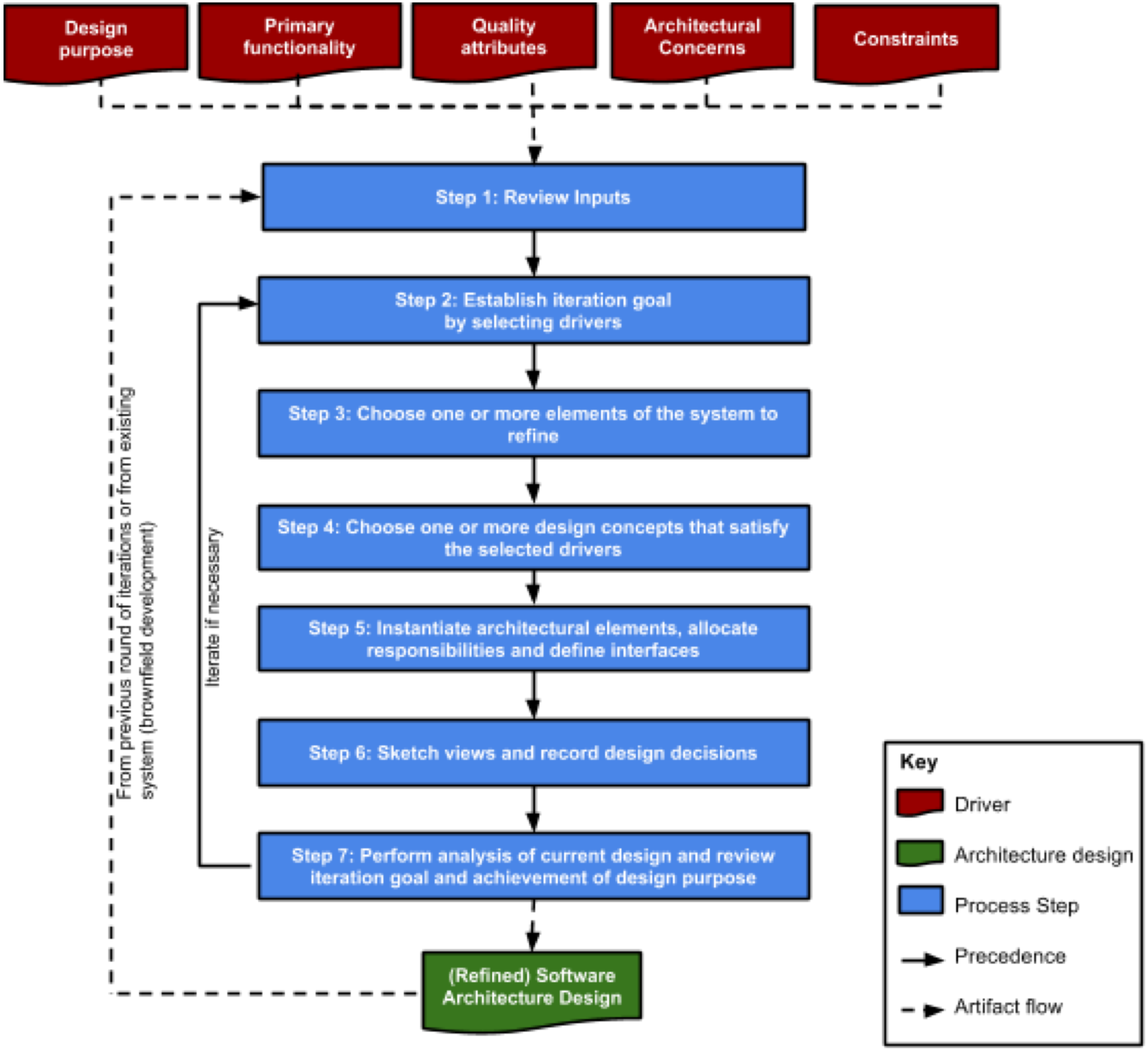
The Need for a Method

- How do you actually perform design?
- Performing design to *ensure* that the drivers are satisfied requires a principled method.
  - By *principled*, we mean a method that takes into account all of the relevant aspects needed to produce an adequate design.
- A method provides guidance.

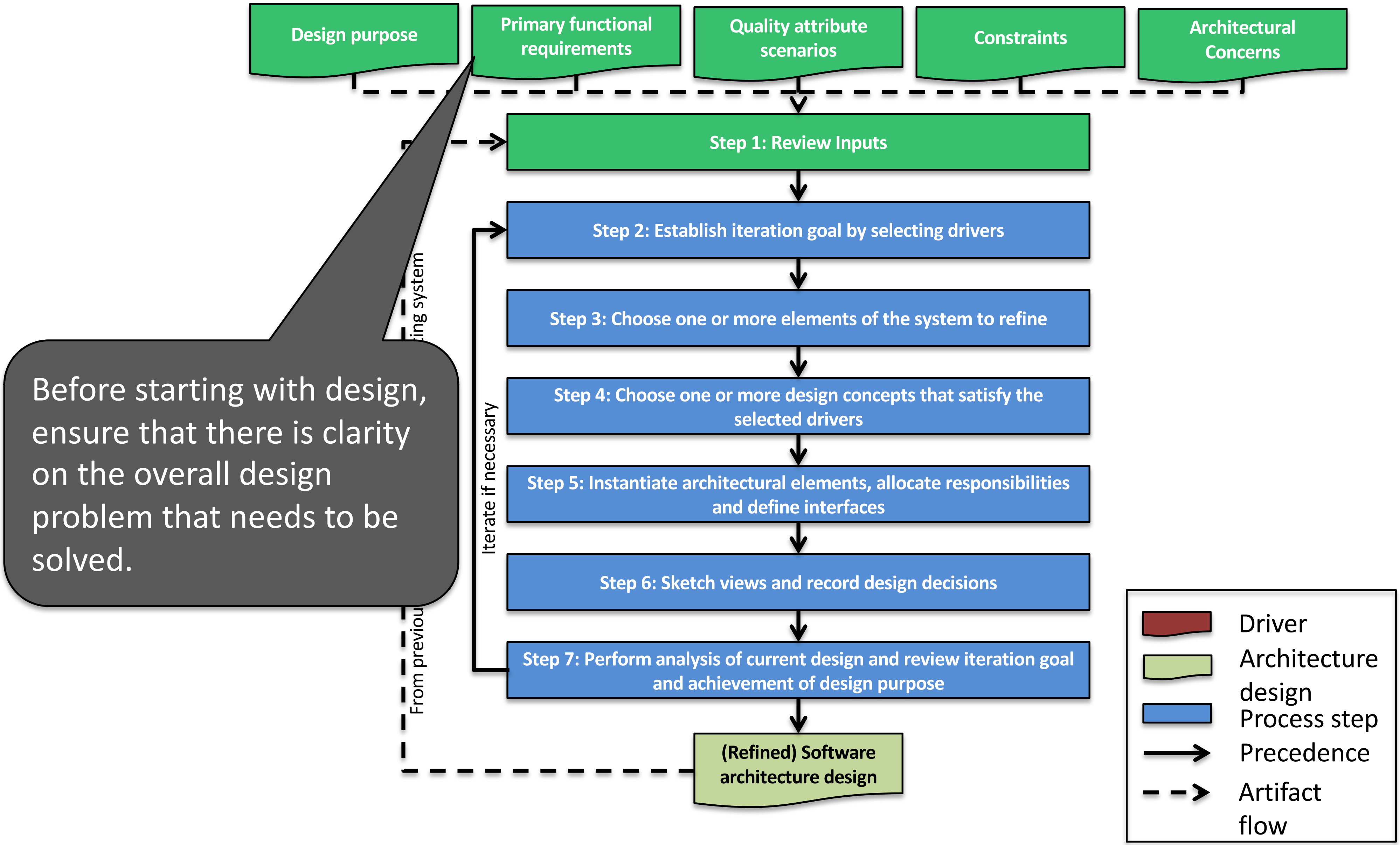
Architecture Driven Design (ADD)

- In ADD architectural design is performed in a series of *rounds*.
  - Each design round may take place within a project increment such as a sprint.
- Within these rounds, a series of *design iterations* are performed.
- Attribute driven design (ADD) provides detailed, step-by-step guidance on the tasks to be performed inside the iterations.

ADD's Steps

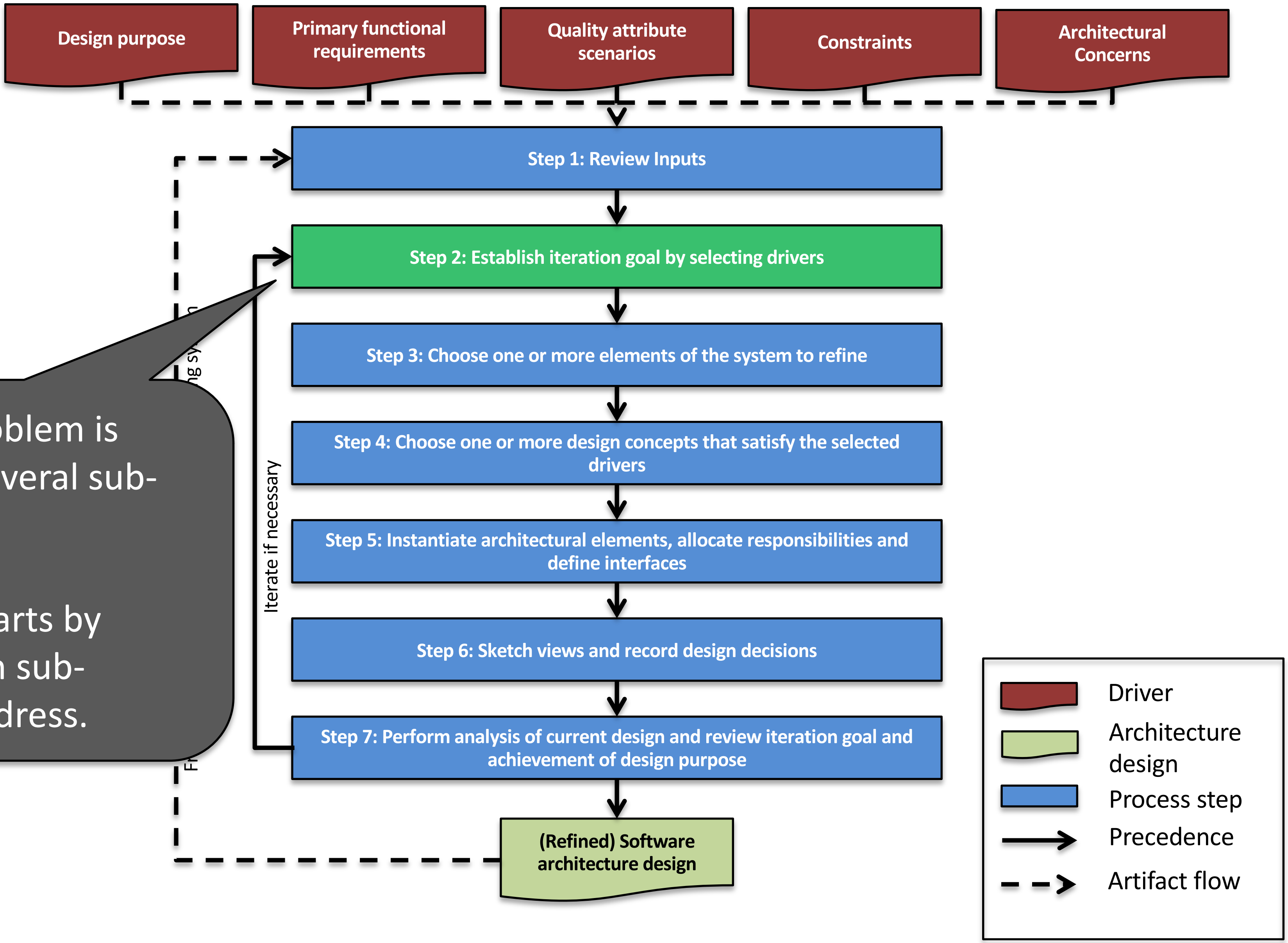

Design purpose
Primary functionality
Quality attributes
Architectural Concerns
Constraints
Step 1: Review Inputs
Step 2: Establish iteration goal by selecting drivers
Step 3: Choose one or more elements of the system to refine
Step 4: Choose one or more design concepts that satisfy the selected drivers
Step 5: Instantiate architectural elements, allocate responsibilities and define interfaces
Step 6: Sketch views and record design decisions
Step 7: Perform analysis of current design and review iteration goal and achievement of design purpose
(Refined) Software Architecture Design
From previous round of iterations or from existing system (brownfield development)
Iterate if necessary
Key
Driver
Architecture design
Process Step
Precedence
Artifact flow

Step 1: Review Inputs


Design purpose
Primary functional requirements
Quality attribute scenarios
Constraints
Architectural Concerns
Step 1: Review Inputs
Step 2: Establish iteration goal by selecting drivers
Step 3: Choose one or more elements of the system to refine
Step 4: Choose one or more design concepts that satisfy the selected drivers
Step 5: Instantiate architectural elements, allocate responsibilities and define interfaces
Step 6: Sketch views and record design decisions
Step 7: Perform analysis of current design and review iteration goal and achievement of design purpose
(Refined) Software architecture design
Iterate if necessary
From previou
ing system
🟥 Driver
🟩 Architecture design
🟦 Process step
→ Precedence
- - → Artifact flow

Before starting with design, ensure that there is clarity on the overall design problem that needs to be solved.

Step 2: Establish Iteration Goal



Design purpose

Primary functional requirements

Quality attribute scenarios

Constraints

Architectural Concerns

Step 1: Review Inputs

Step 2: Establish iteration goal by selecting drivers

Step 3: Choose one or more elements of the system to refine

Step 4: Choose one or more design concepts that satisfy the selected drivers

Step 5: Instantiate architectural elements, allocate responsibilities and define interfaces

Step 6: Sketch views and record design decisions

Step 7: Perform analysis of current design and review iteration goal and achievement of design purpose

(Refined) Software architecture design

🟥 Driver

🟩 Architecture design

🟦 Process step

→ Precedence

- - > Artifact flow

The design problem is divided into several sub-problems.

An iteration starts by deciding which sub-problem to address.

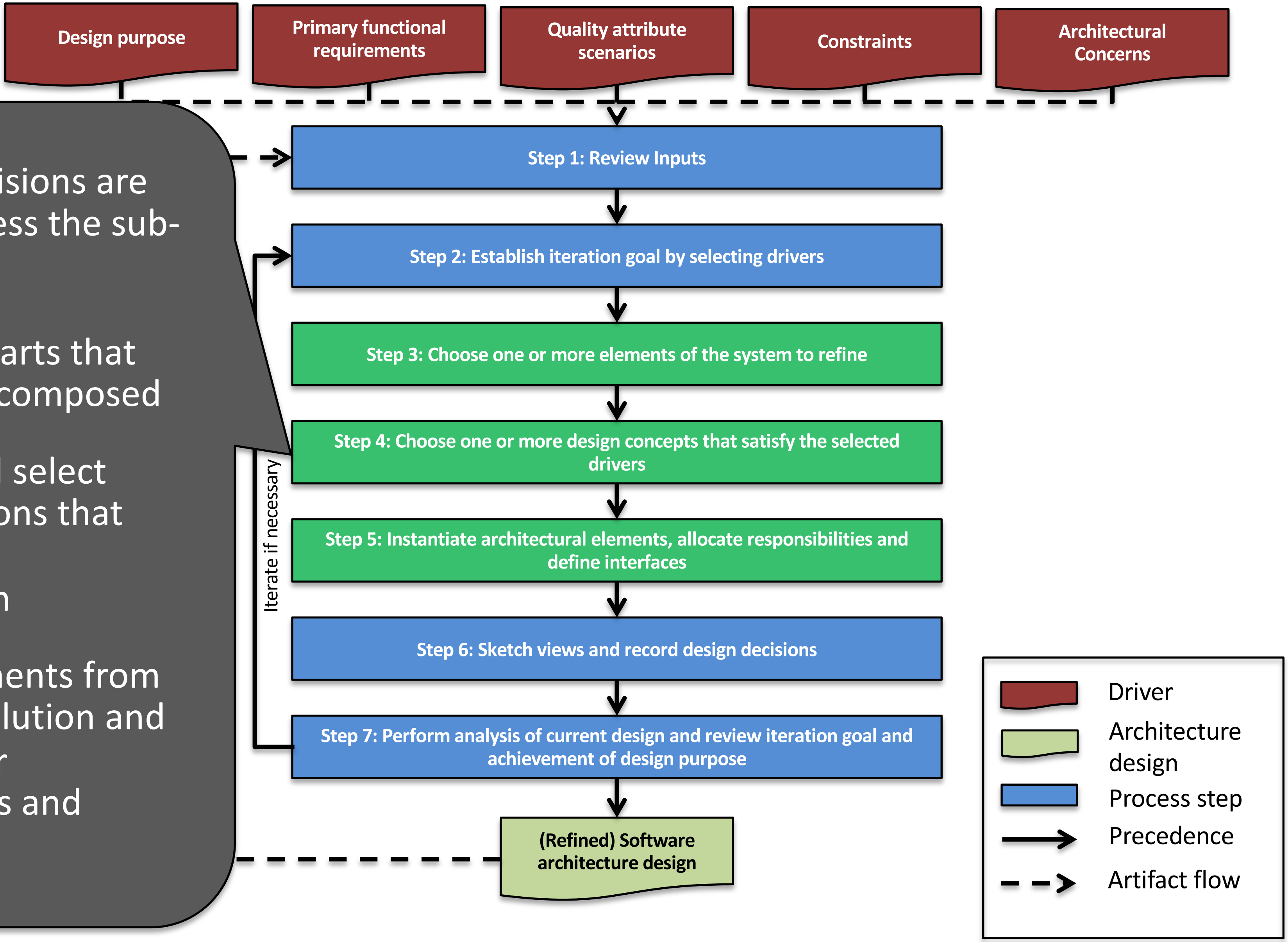
Steps 3-5: Choose and Instantiate Elements

3 types of decisions are made to address the sub-problem:

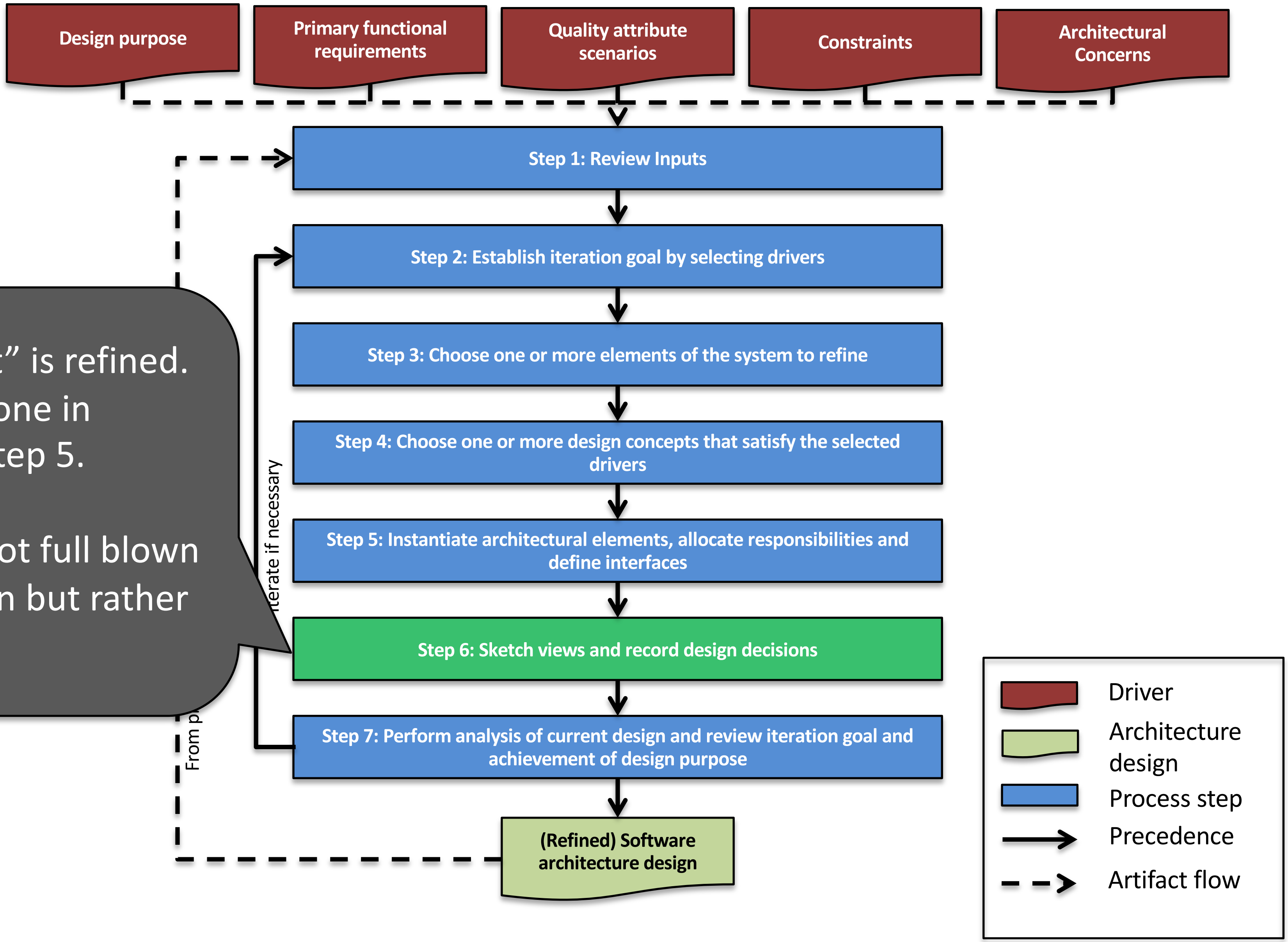
1. Select the parts that need to be decomposed

2. Identify and select existing solutions that support the decomposition

3. Create elements from the existing solution and establish their responsibilities and interfaces


Design purpose
Primary functional requirements
Quality attribute scenarios
Constraints
Architectural Concerns
Step 1: Review Inputs
Step 2: Establish iteration goal by selecting drivers
Step 3: Choose one or more elements of the system to refine
Step 4: Choose one or more design concepts that satisfy the selected drivers
Step 5: Instantiate architectural elements, allocate responsibilities and define interfaces
Step 6: Sketch views and record design decisions
Step 7: Perform analysis of current design and review iteration goal and achievement of design purpose
Iterate if necessary
(Refined) Software architecture design
Driver
Architecture design
Process step
Precedence
Artifact flow

Step 6: Sketch Views



Design purpose

Primary functional requirements

Quality attribute scenarios

Constraints

Architectural Concerns

Step 1: Review Inputs

Step 2: Establish iteration goal by selecting drivers

Step 3: Choose one or more elements of the system to refine

Step 4: Choose one or more design concepts that satisfy the selected drivers

Step 5: Instantiate architectural elements, allocate responsibilities and define interfaces

Step 6: Sketch views and record design decisions

Step 7: Perform analysis of current design and review iteration goal and achievement of design purpose

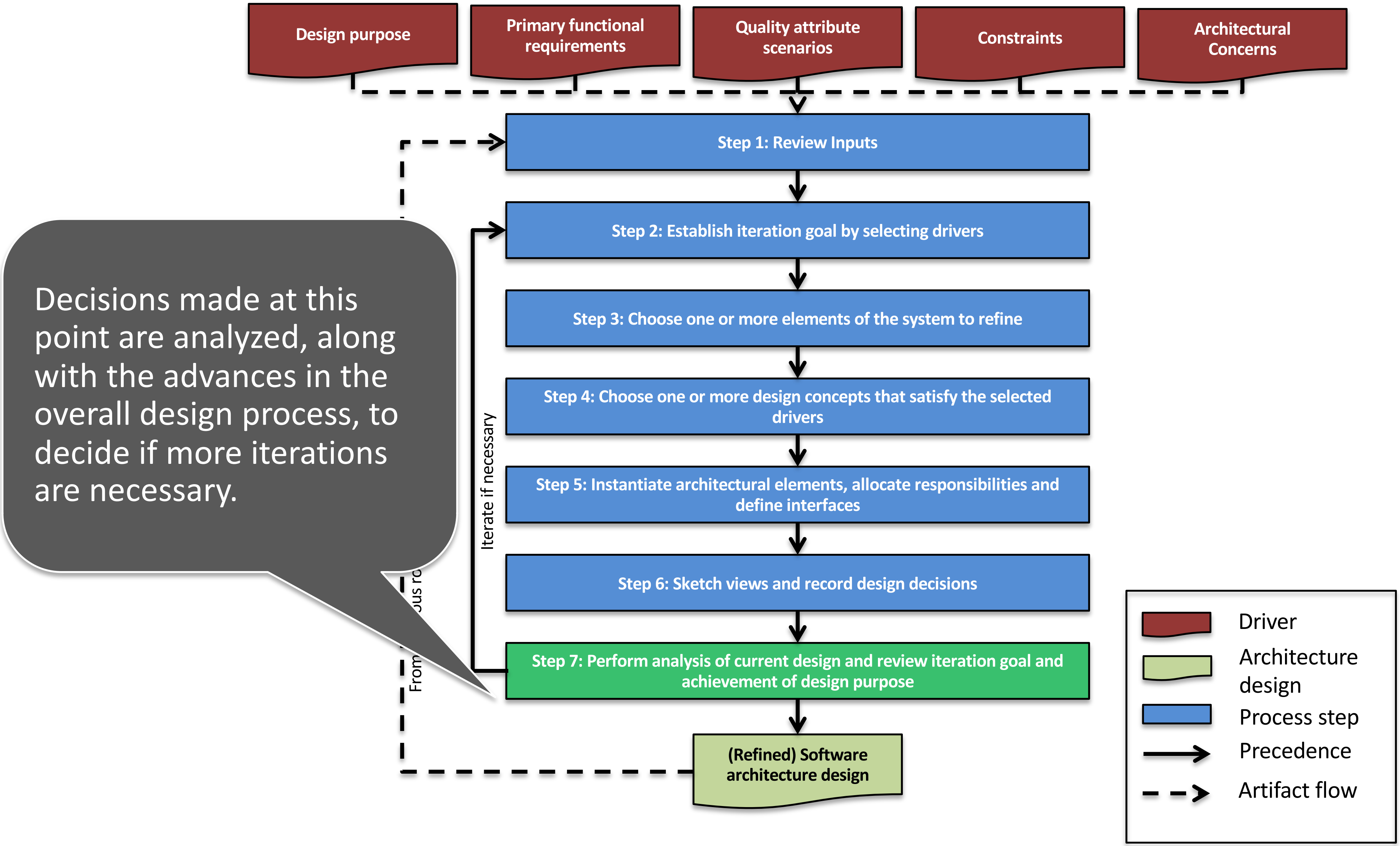
(Refined) Software architecture design

- Driver
- Architecture design
- Process step
- Precedence
- Artifact flow

The “blueprint” is refined. This may be done in parallel with step 5.

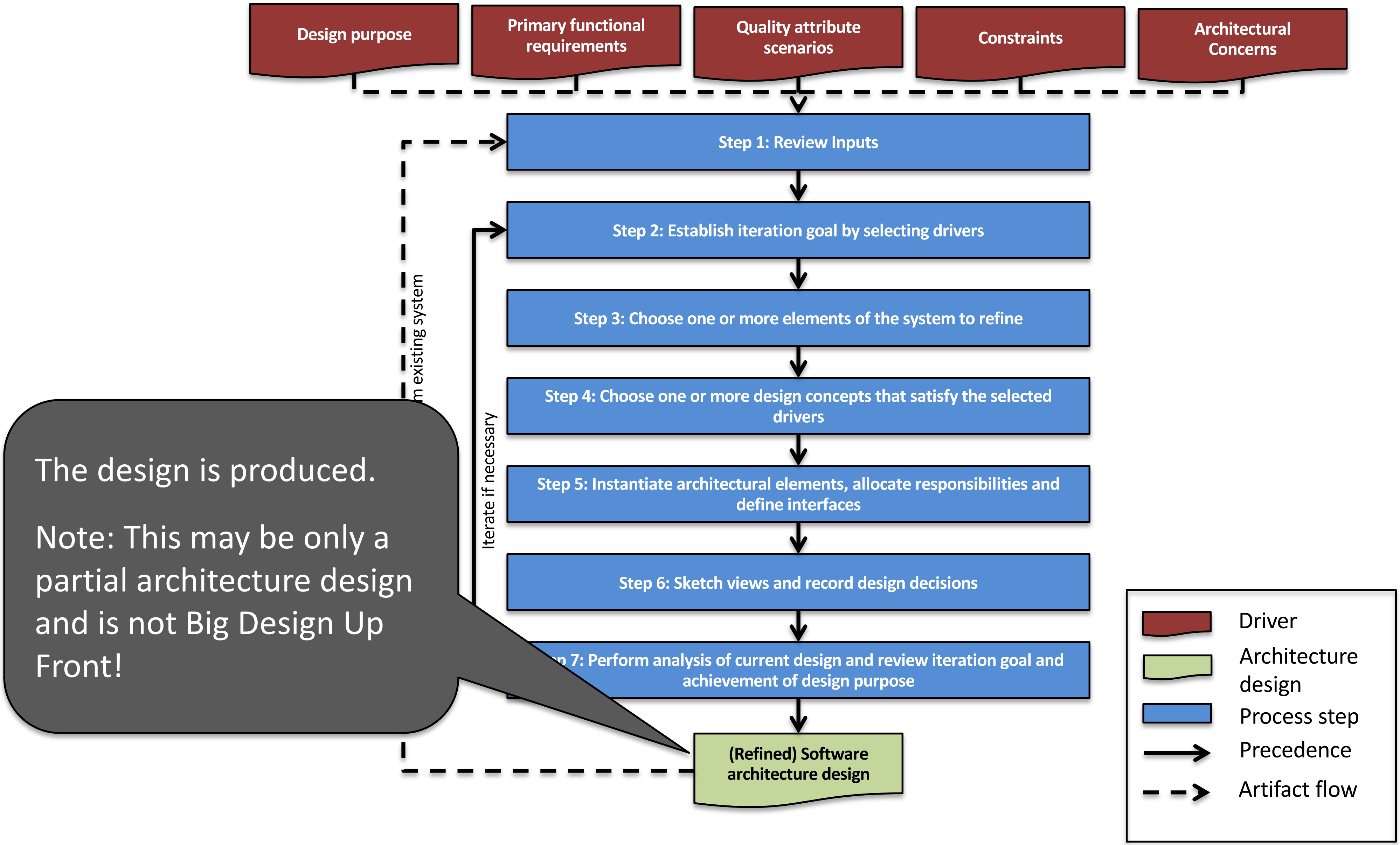
Note: This is not full blown documentation but rather *sketches*.

Step 7: Perform Analysis


Design purpose
Primary functional requirements
Quality attribute scenarios
Constraints
Architectural Concerns
Step 1: Review Inputs
Step 2: Establish iteration goal by selecting drivers
Step 3: Choose one or more elements of the system to refine
Step 4: Choose one or more design concepts that satisfy the selected drivers
Step 5: Instantiate architectural elements, allocate responsibilities and define interfaces
Step 6: Sketch views and record design decisions
Step 7: Perform analysis of current design and review iteration goal and achievement of design purpose
(Refined) Software architecture design
Iterate if necessary
From
ous ro
Driver
Architecture design
Process step
Precedence
Artifact flow

Decisions made at this point are analyzed, along with the advances in the overall design process, to decide if more iterations are necessary.

ADD Output/Iteration



Design purpose

Primary functional requirements

Quality attribute scenarios

Constraints

Architectural Concerns

Step 1: Review Inputs

Step 2: Establish iteration goal by selecting drivers

Step 3: Choose one or more elements of the system to refine

Step 4: Choose one or more design concepts that satisfy the selected drivers

Step 5: Instantiate architectural elements, allocate responsibilities and define interfaces

Step 6: Sketch views and record design decisions

p 7: Perform analysis of current design and review iteration goal and achievement of design purpose

(Refined) Software architecture design

Driver

Architecture design

Process step

Precedence

Artifact flow

The design is produced.

Note: This may be only a partial architecture design and is not Big Design Up Front!

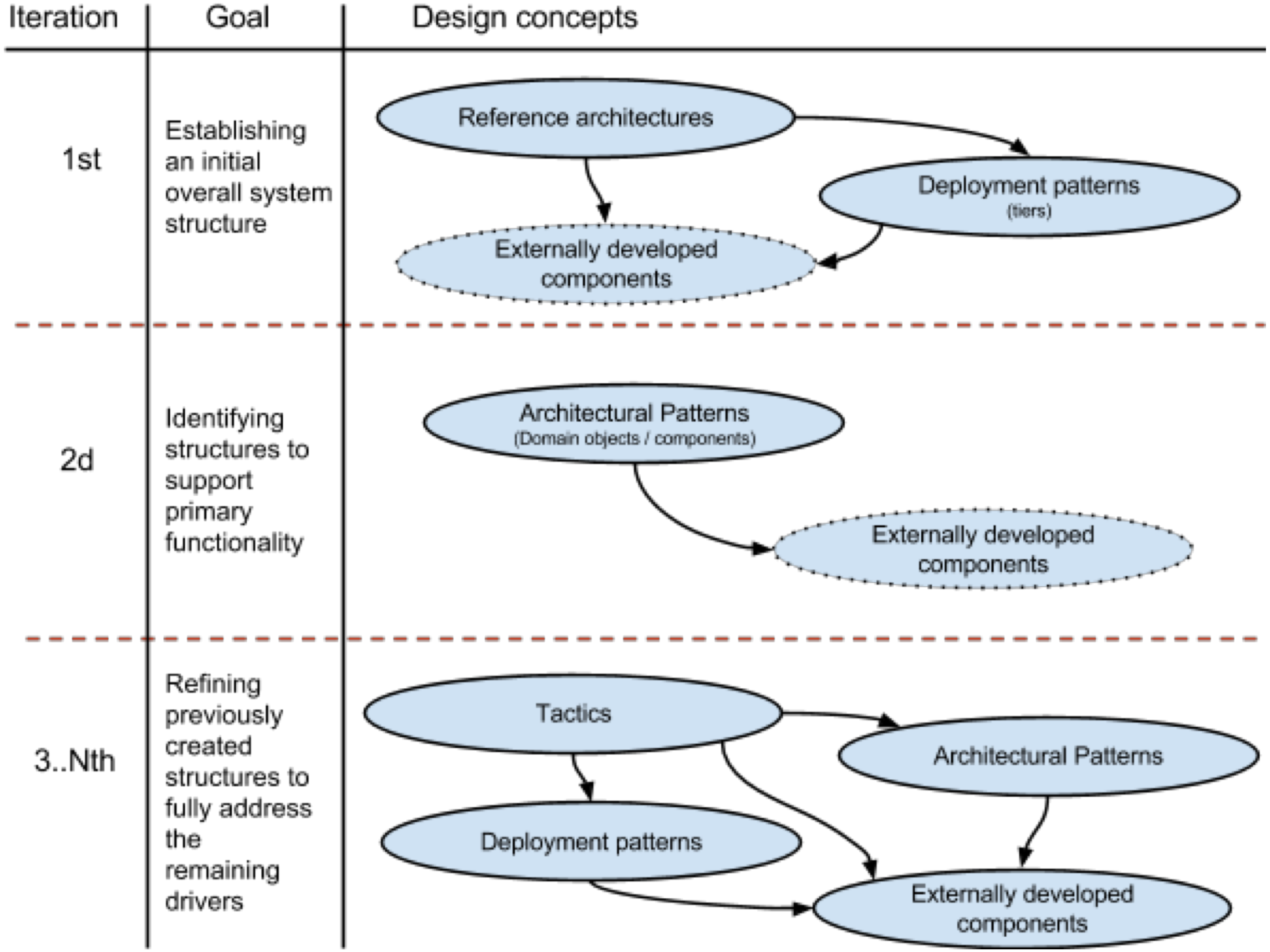
Following a Design Roadmap

- Design of software systems falls into three broad categories:

1. the design of a greenfield system for a mature (i.e. well known) domain,
2. the design of a greenfield system for a domain that is novel (that is, a domain which has a less established infrastructure and knowledge base), and
3. the design for making changes to an existing system.

- Each of these involves a different *roadmap*: the steps you should perform in the iterations.

Design of greenfield systems for mature domains



Iteration	Goal	Design concepts
1st	Establishing an initial overall system structure	Reference architectures Deployment patterns (tiers) Externally developed components
2d	Identifying structures to support primary functionality	Architectural Patterns (Domain objects / components) Externally developed components
3..Nth	Refining previously created structures to fully address the remaining drivers	Tactics Architectural Patterns Deployment patterns Externally developed components

Design of greenfield systems for novel domains

- In novel domains, it is harder to establish a roadmap:
  - there may not be reference architectures
  - there may be few externally developed components.
- You will need to work from first principles.
- Even in this case, design concepts like tactics and patterns can still guide you, aided by prototyping.
- Your iteration goals will mostly be to refine previously created structures to address the drivers.

Design for an existing system (brownfield)

- For an existing system you may need to satisfy new requirements or correct issues
  - this may catalyze changes to the architecture
- Or you may be making architectural changes to refactor the system to fix quality attribute problems
  - e.g. the system is slow, or frequently crashes, or is hard to modify.

Design for an existing system (brownfield)

- If you understand your architecture you can perform design as with greenfield systems (after the initial design iteration).
- Your iteration goals are to identify and refine structures to satisfy architectural drivers:
  - new functionality
  - quality attributes
  - architectural concerns
- This will not involve establishing a new overall system structure, unless you are doing a major refactoring.

Identifying Design Concepts

- There are dozens of patterns and components you could use to address any issue.
- To make things worse, these design concepts are scattered across many sources—popular press, research literature, books, the internet—and described differently.
- Finally, once you have identified design alternatives, you need to *select* among them.

Selecting Design Concepts

- Once you have identified a list of design concepts, you need to select among them.
- How?
- Create a table that lists the pros and cons of each alternative and select based on your criteria and drivers.

Example Table to Support Selection

Name of alternative	**Pros**	**Cons**	**Cost**
Web application	• Can be accessed from a variety of platforms using a standard web browser • Fast page loading • Simple deployment	• Does not support “rich” interaction	Low
Rich internet application	• Supports “rich” user interaction • Simple deployment and updating	• Longer page loading times • Requires a runtime environment to be installed on the client browser	Medium
Mobile application	• Supports “rich” user interaction	• Less portability • Screen limitations	High

Selecting Design Concepts

- If the table is not enough you may need to create prototypes and collect measurements.
- But creating prototypes can be costly compared to analysis.
- What to do, and why?

To Prototype or Not to Prototype?

Questions to consider:

- Does the project incorporate emerging technologies?
- Is the technology new in the company?
- Are there drivers, particularly quality attributes, whose satisfaction using the selected technology presents risks?
- Is there a lack of evidence that the selected technology will help satisfy the project drivers?
- Are there configuration options associated with the technology that need to be tested or understood?
- Is it unclear whether the selected technology can be integrated with the project?

If most of your answers are “yes” you should consider a throwaway prototype.

Producing Structures

- Design concepts won’t help you satisfy your drivers unless you produce *structures*
  - you need to identify and connect elements that are derived from the selected design concepts.
- This is “instantiation” in ADD: creating elements and relationships, and associating responsibilities with these elements

Producing Structures

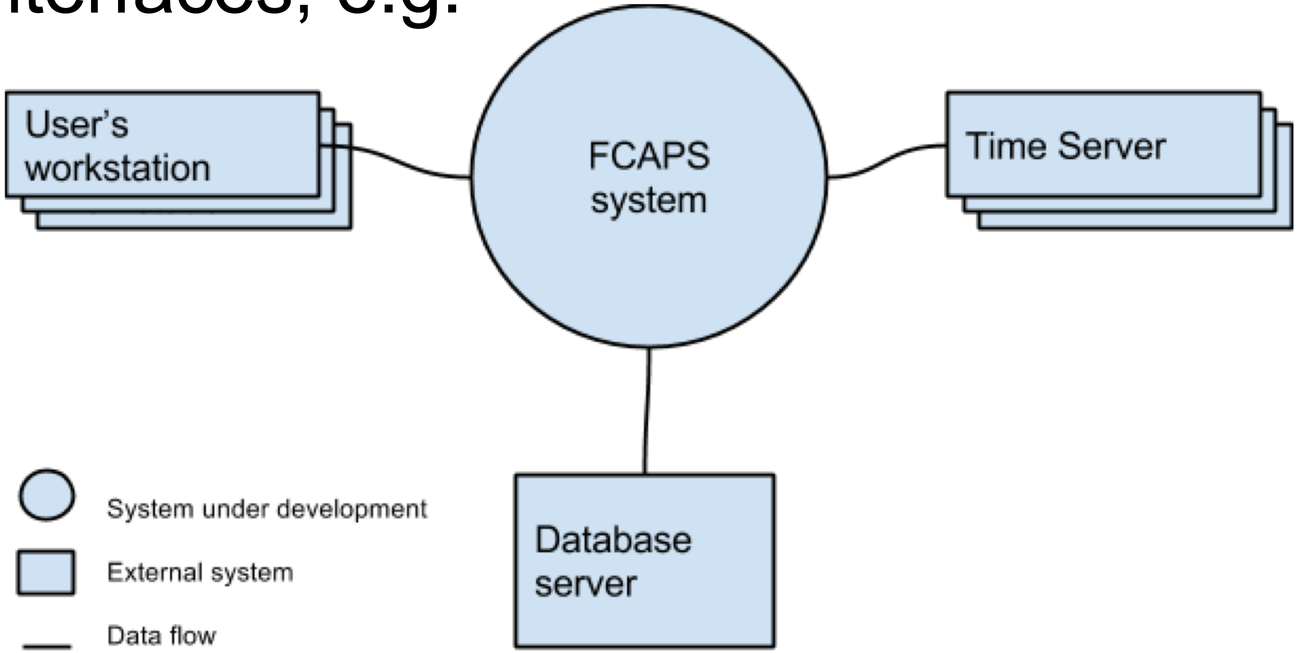
- Structures are categorized as:
  - Module structures: elements that exist at development time, like files, modules and classes.
  - Component and Connector (C&C) structures: dynamic elements that exist at run-time, like processes and threads.
  - Allocation structures: software elements (from a module or C&C structure) and non-software elements, like file systems, hardware, and development teams.
- When you instantiate a design concept you may need to produce more than one structure.

Defining Interfaces

- Interfaces are the externally visible properties of elements which establish a contractual specification that allows elements to collaborate and exchange information.
- There are two categories of interfaces: external and internal.

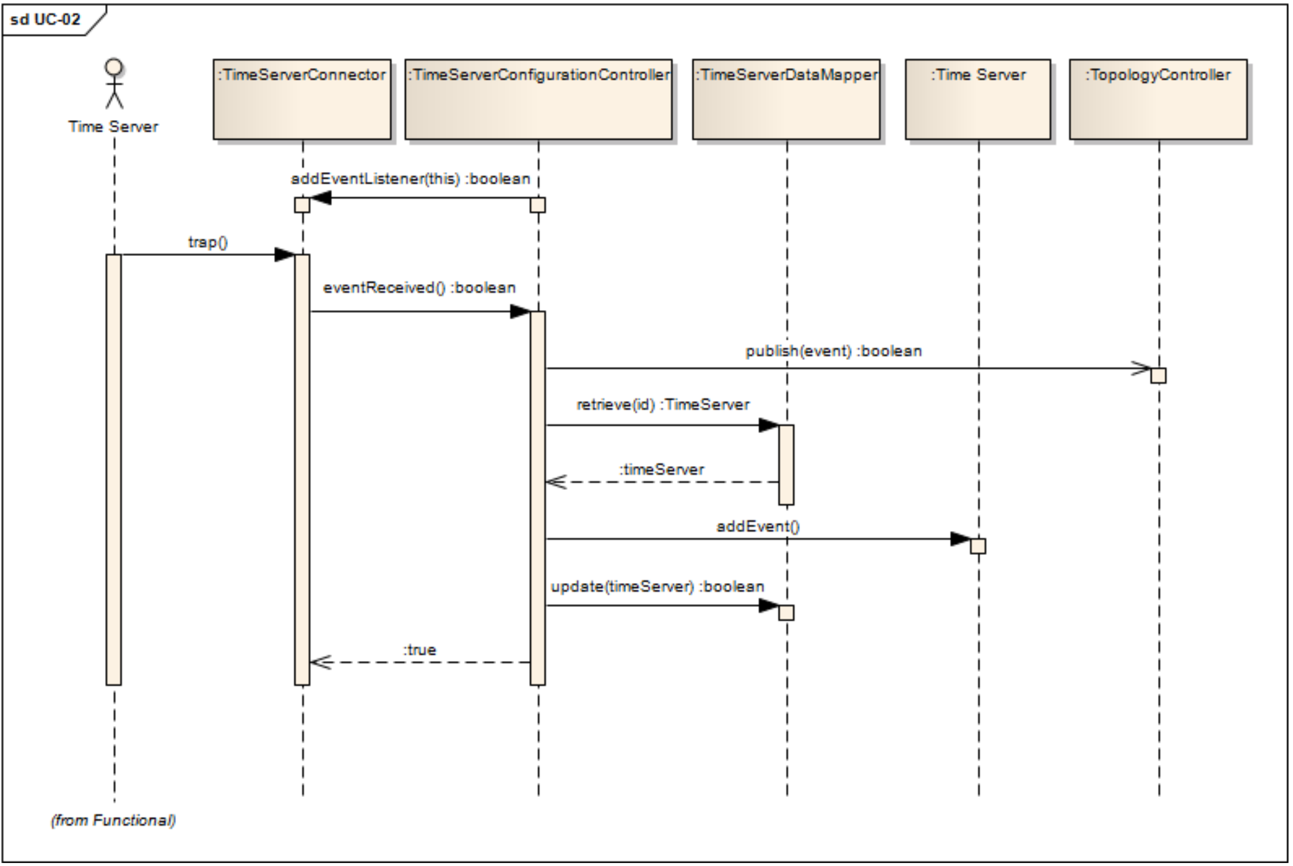
External Interfaces

- External interfaces include interfaces from other systems that are *required* by the system that you are developing and interfaces that are *provided* by your system to other systems.
- A system context diagram can help understand external interfaces, e.g.



Internal Interfaces

- Internal interfaces are interfaces between the elements that result from the instantiation of design concepts.
- A sequence diagram can help understand internal interfaces, e.g



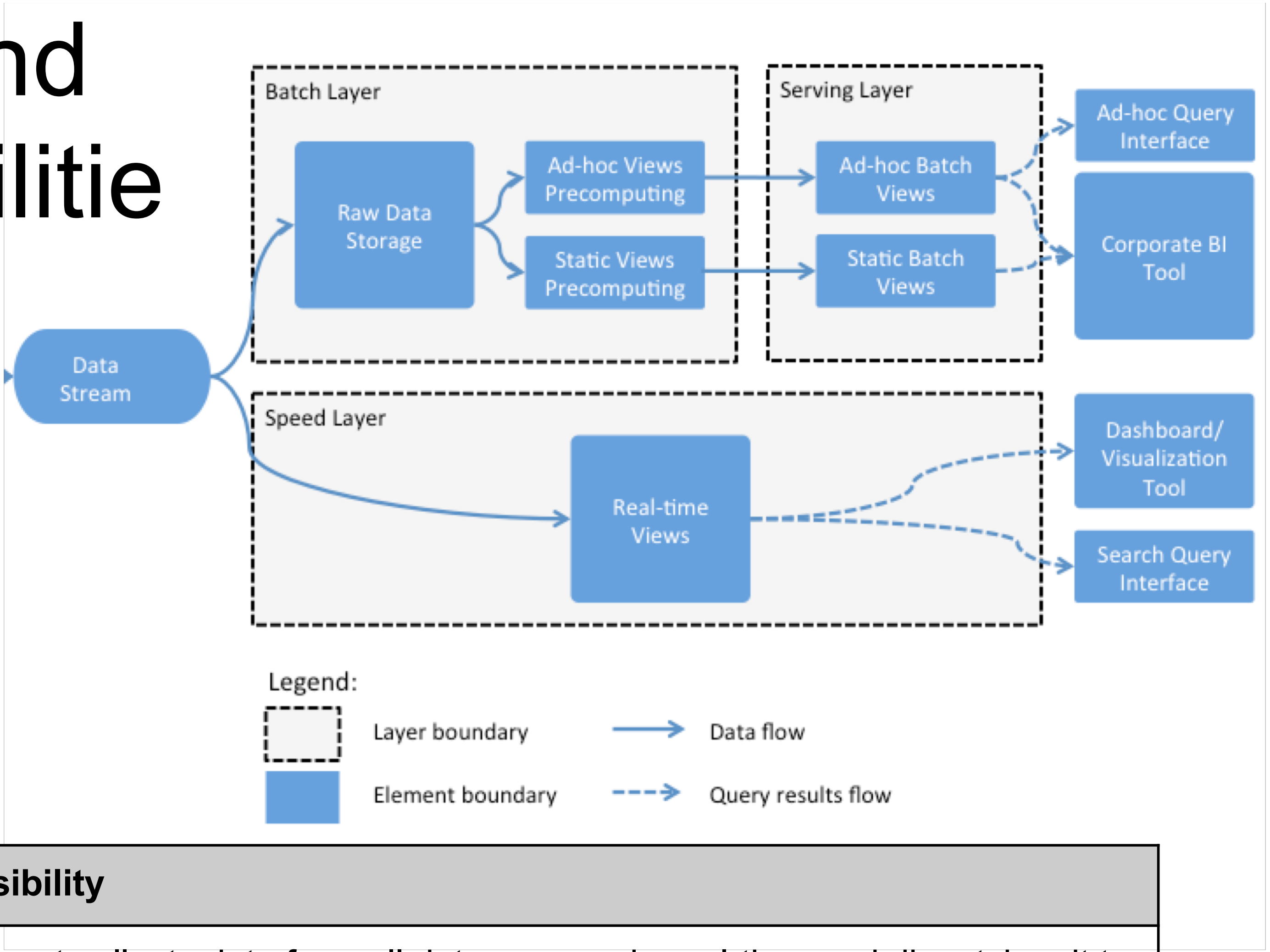
Documenting During Design

- A software architecture is typically documented as a set of *views*, which represent the different structures that compose the architecture.
- The formal documentation of views is not part of the design process.
- Structures, however, are produced as part of design.
- These should be captured, even if they are informal (i.e. sketches).

Recording Sketches

- If you use an informal notation for sketches, you should maintain consistency in the use of symbols (and add a legend to your diagrams!).
- You should develop a discipline of writing down the responsibilities that you allocate to the elements as you create the structures.
- Writing it down *at that moment* ensures that you won’t have to remember it later.

Example Sketch and Responsibilitie s



Element	Responsibility
Data Stream	This element collects data from all data sources in real-time and dispatches it to both the Batch Layer and the Speed Layer for processing.
Batch Layer	This layer is responsible for storing raw data and pre-computing the Batch Views to be stored in the Serving Layer.
…	…

Recording Design Decisions

- In each design iteration, you make important design decisions to achieve your iteration goal. These design decisions include:
  - Selecting a design concept from several alternatives
  - Creating structures by instantiating the selected design concept
  - Establishing relationships between elements and defining interfaces
  - Allocating resources (people, hardware, computation, etc.)
  - Others

Recording Design Rationale

- Recording design decisions *beyond* the elements, relationships, and properties is fundamental to help others understand how you arrived at the result: the *design rationale*.

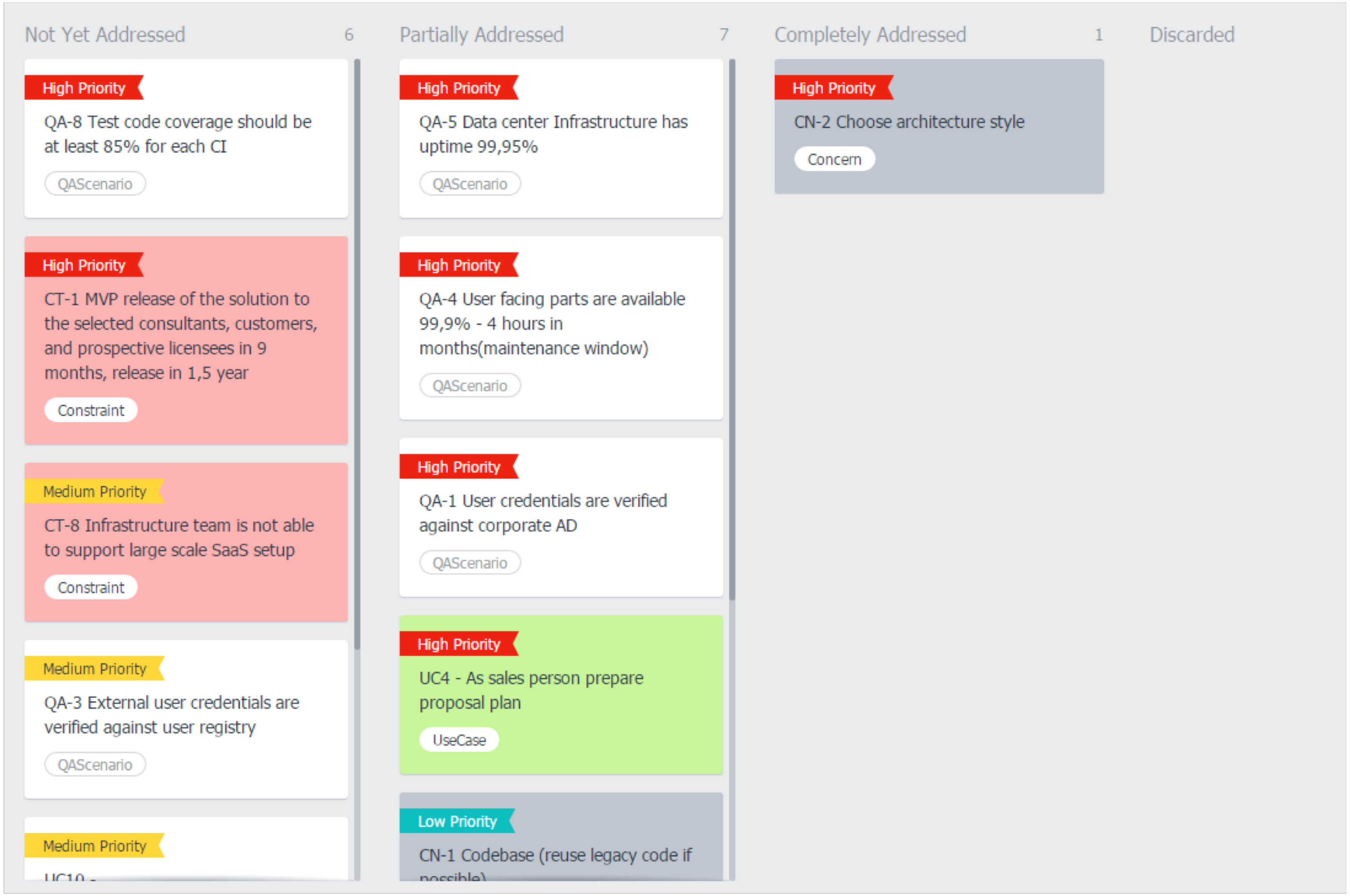
Recording Design Rationale

- Some information that can be useful to record includes:
  - What evidence was produced to justify decisions?
  - Who did what?
  - Why were shortcuts taken?
  - Why were tradeoffs made?
  - What assumptions did you make?

Tracking Design Progress

- When you are performing design, however, there are several questions that you want to answer:
  - How much design do we need to do?
  - How much design has been done so far?
  - Are we finished?
- Agile practices such as the use of backlogs and kanban boards can help you track the design progress and answer these questions.
- Any development project using any methodology, should track progress.

Example Kanban Board



Not Yet Addressed 6

High Priority
QA-8 Test code coverage should be at least 85% for each CI
QAScenario

High Priority
CT-1 MVP release of the solution to the selected consultants, customers, and prospective licensees in 9 months, release in 1,5 year
Constraint

Medium Priority
CT-8 Infrastructure team is not able to support large scale SaaS setup
Constraint

Medium Priority
QA-3 External user credentials are verified against user registry
QAScenario

Medium Priority
UC10 -

Partially Addressed 7

High Priority
QA-5 Data center Infrastructure has uptime 99,95%
QAScenario

High Priority
QA-4 User facing parts are available 99,9% - 4 hours in months(maintenance window)
QAScenario

High Priority
QA-1 User credentials are verified against corporate AD
QAScenario

High Priority
UC4 - As sales person prepare proposal plan
UseCase

Low Priority
CN-1 Codebase (reuse legacy code if possible)

Completely Addressed 1

High Priority
CN-2 Choose architecture style
Concern

Discarded

Summary

- We presented a detailed walk-through of the Attribute Driven Design method, version 3.0.
- We discussed additional aspects that need to be considered in the steps of the design process:
  - the use of a backlog,
  - the different possible design roadmaps (for greenfield, brownfield, and novel contexts),
  - the identification and selection of design concepts and their use in producing structures,
  - the definition of interfaces, and
  - the production of preliminary documentation.
- We stress that preliminary documentation and analysis activities need to be regularly performed as integral parts of the design process.