Common Modeling Mistakes and the Necessity of Declarative Modeling
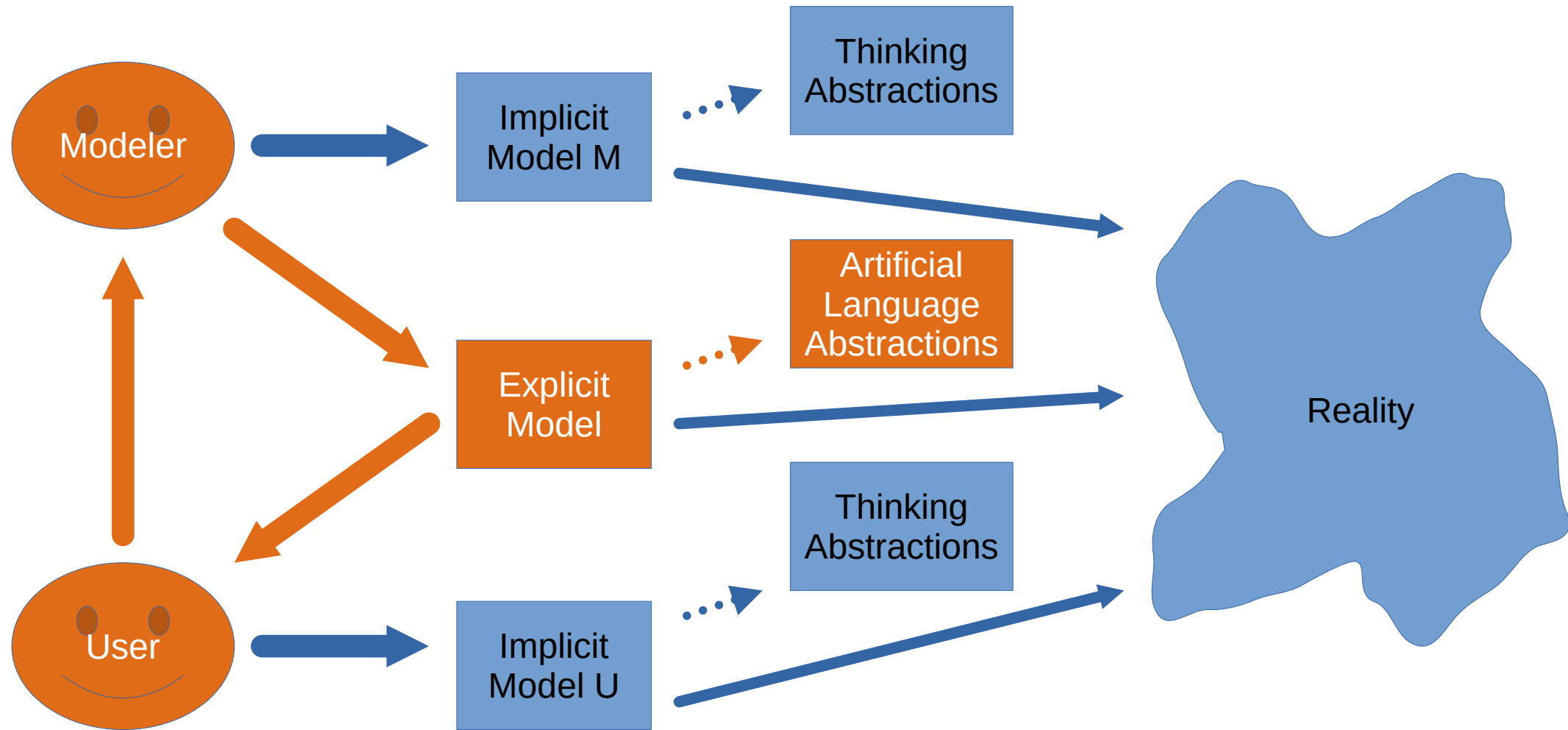
Wim Bast / Modeling Value Group / DClare

LangDev CON 2024

Modeling

- Each model is based on some (implicit) perception of the (user)domain.
- The mapping of a model to the reality can never be explicit.
- Using the right abstractions is essential, hence use the appropriate artificial language.
- An understandable model does not imply that you understand the reality, it can be well be an over-simplification of reality.
- How do we know a model is good enough?

Abstractions Languages

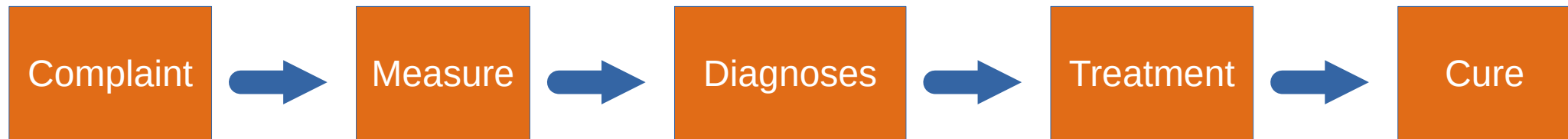| Declarative		Imperative	
Class Instantiation Specialization Containment Property Relation Function Rule State Knowledge	Focus on Goal Integratable Functional Natural Complete More Verifiable Minimal Complexity Classification Based Less Intuitive	Process Data Method Control Flow Data Flow Thread Time Order Event MVC	Focus on How Non Integratable Technical Artificial Over Specified Less Verifiable Over Simplified Example Based More Intuitive

Over Specification

Over Simplification

| Complaint	→	Measure	→	Diagnoses	→	Treatment	→	Cure

Action

Imperative Modeling

- Does the imposed (process)restriction really works in reality?
- Data/Process thinking is a technical abstraction.
- Imperative thinking implies that something or someone is in control, imposing a process.
- A process always implies a context (that is mostly undefined).
- Imperative modeling has the tendency to both over-simplifying and over-specifying the domain.
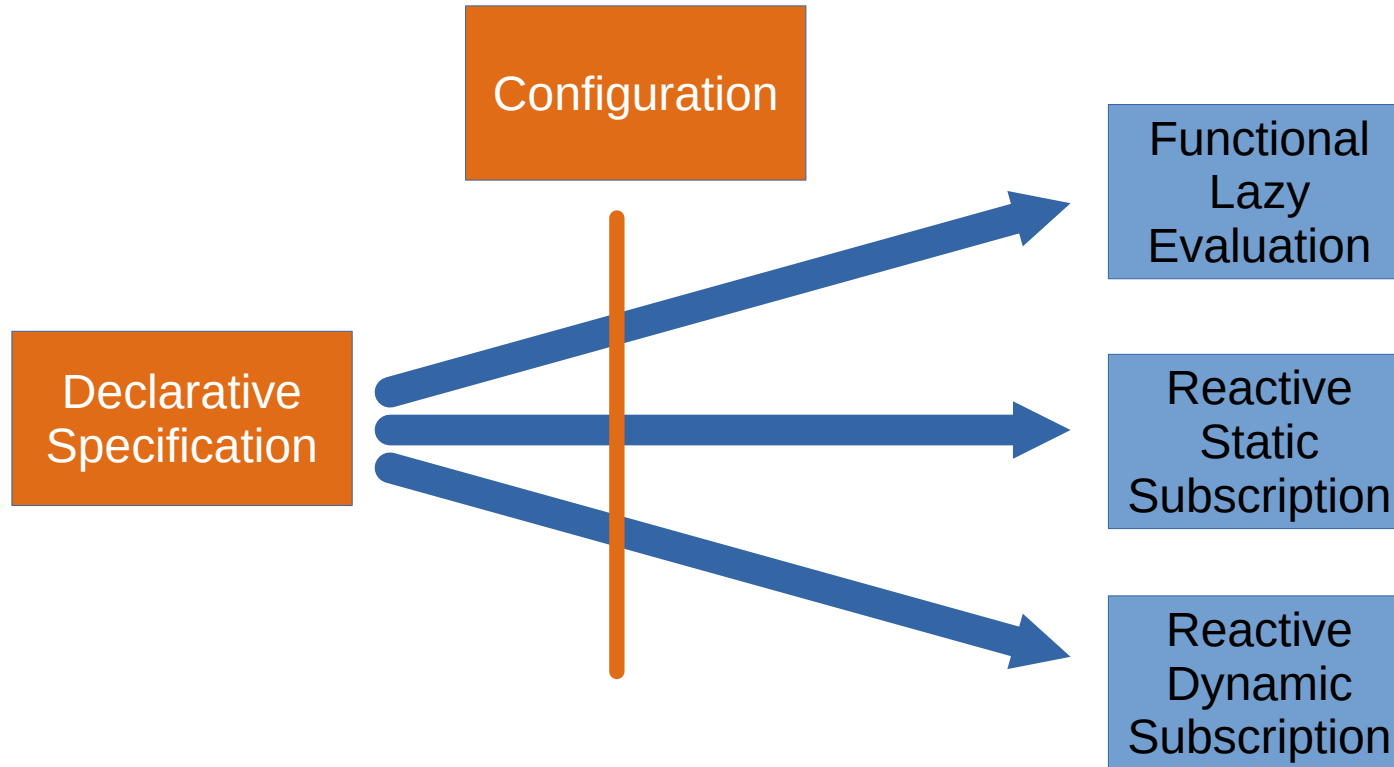- Processes mostly assume a 'flat' world (no many-relations).
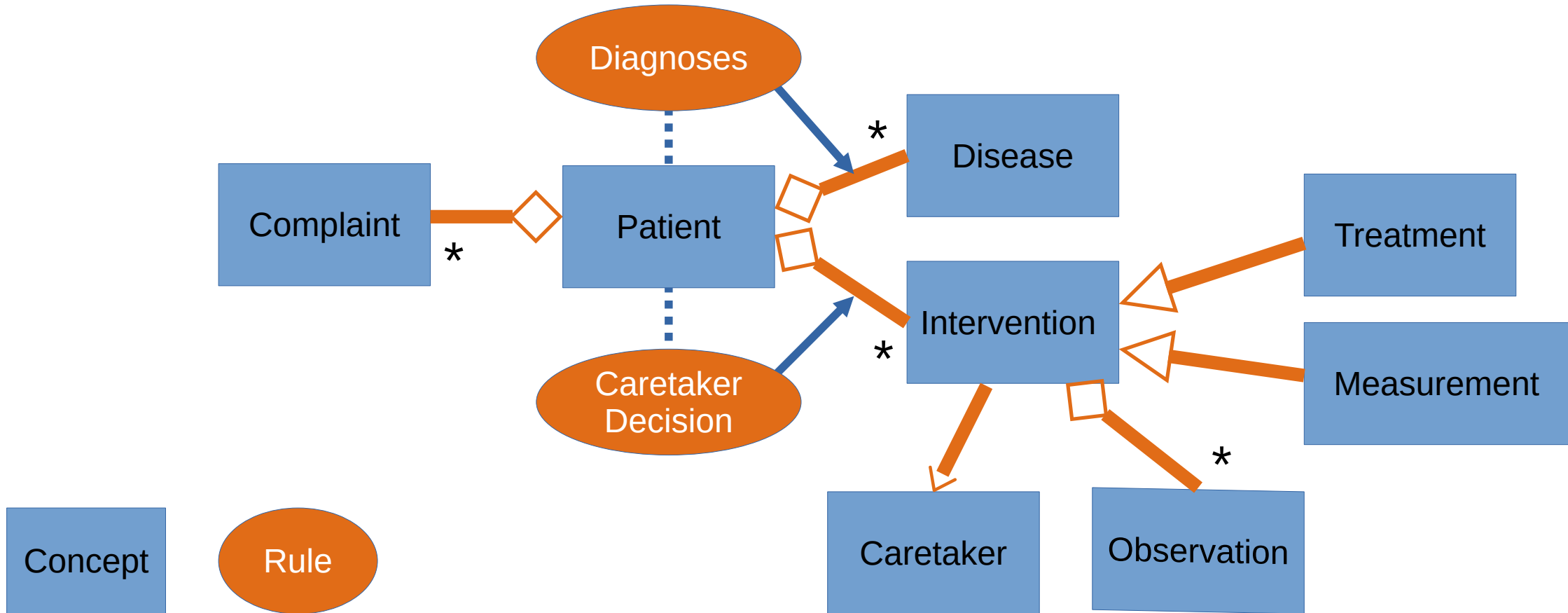
Declarative Modeling

- Declarative modeling helps you to define a model with minimal complexity of the domain, without over-specification.

- Class and relations modeling is absolutely necessary.

- Declarative Rules define the knowledge about the domain.

- You need to define the rules to be sure you are doing the correct thing.

- Processes can be derived from the combined class/rule model, hence redundant to model by hand.
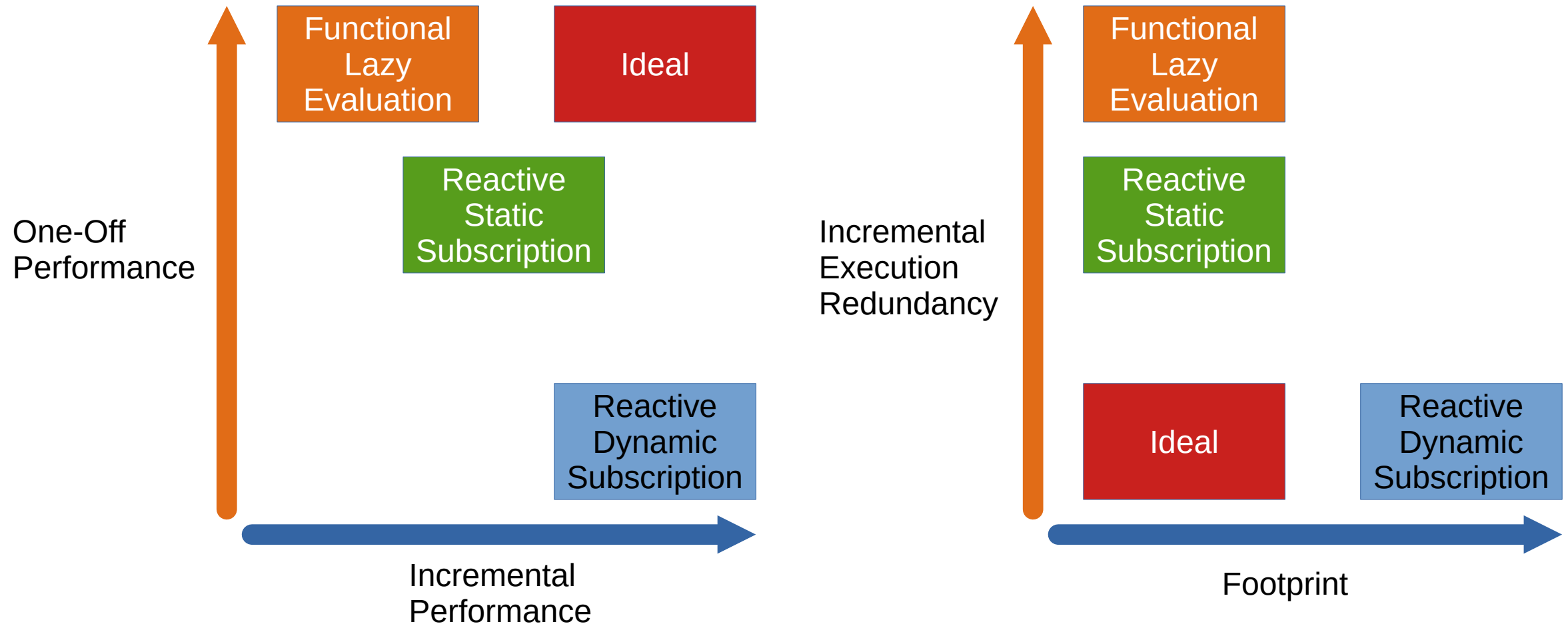
One Declarative Language
Many Execution Strategies

Configuration

Declarative Specification

Functional Lazy Evaluation

Reactive Static Subscription

Reactive Dynamic Subscription

Pros & Cons   Execution Strategies

Discussion