17-723: Designing Large-scale Software Systems

Quality Attributes & Trade-offs

Carnegie Mellon University

This Lecture

- What are Quality Attributes, and why should I care?
- How do I specify Quality Attribute Requirements?
- What are Quality Attribute Trade-offs and where do they occur?

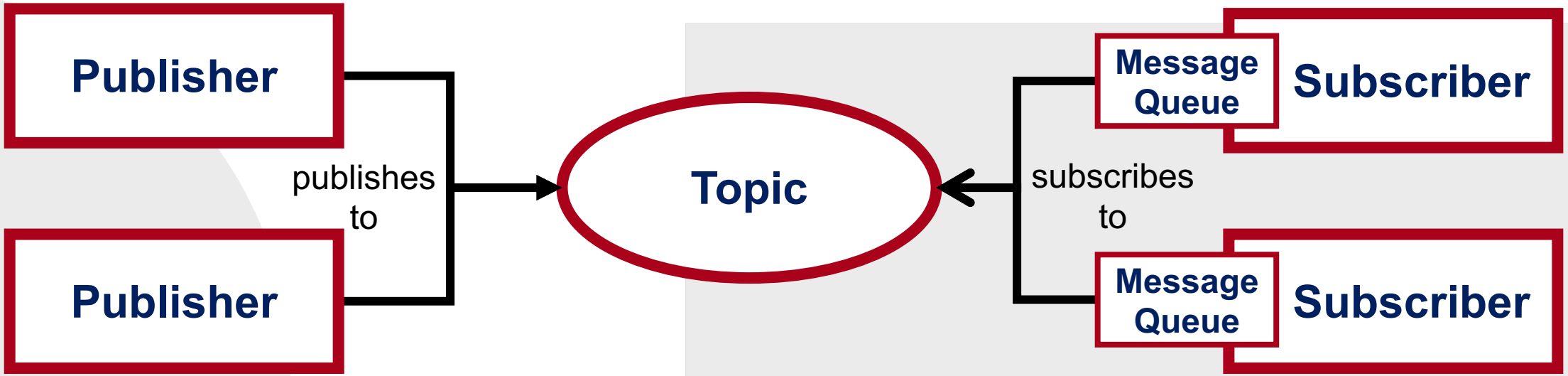
Note: ROS is just a case study, not a learning objective of this lecture

Carnegie Mellon University

Recall from Reading: Publish-Subscribe

```
pub = nh.advertise("topic_name");
pub.publish(msg)
```

```
nh.subscribe("topic_name", 10, callback);
```



Carnegie Mellon University

What are Advantages and Disadvantages of each? Discuss with your Neighbor!

Publish Subscribe	Direct Communication

Carnegie Mellon University

Lesson Learned: Design decisions impact other properties of the system, besides functionality. We call these other properties "**quality attributes**"

What are Advantages and Disadvantages of each?

Publish Subscribe	Direct Communication
Easier to add new publishers and subscribers ➔ **Extensibility**	Sender can detect if listener is not available ➔ **Robustness**
Components can dynamically subscribe and unsubscribe ➔ **Run Time Flexibility**	Easier to understand what components communicate with each other, making it less error-prone ➔ **Understandability**

Carnegie Mellon University

Quality Attributes Measure the "Goodness" of a Design along a Certain Dimension

- Functionality describes **what** the system does, quality attributes describe **how well** it does it
- E.g., **Extensibility**, **Availability**, **Security**, **Performance**, **Robustness**, **Interoperability**, **Testability**, …

Carnegie Mellon University

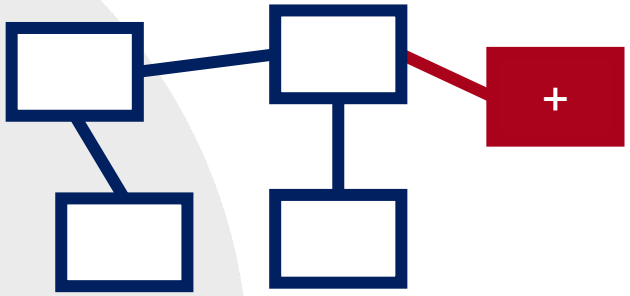
What Quality Attributes are Important for … ?

- Social Media Systems (e.g., Facebook)
- Mars Rovers
- Financial Trading Systems

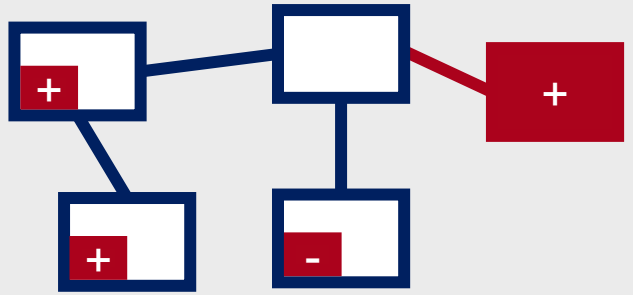
Carnegie Mellon University

Extensibility

The degree to which a system minimizes the amount of **effort** and **error-proneness** of adding **additional functionality**



More Extensible

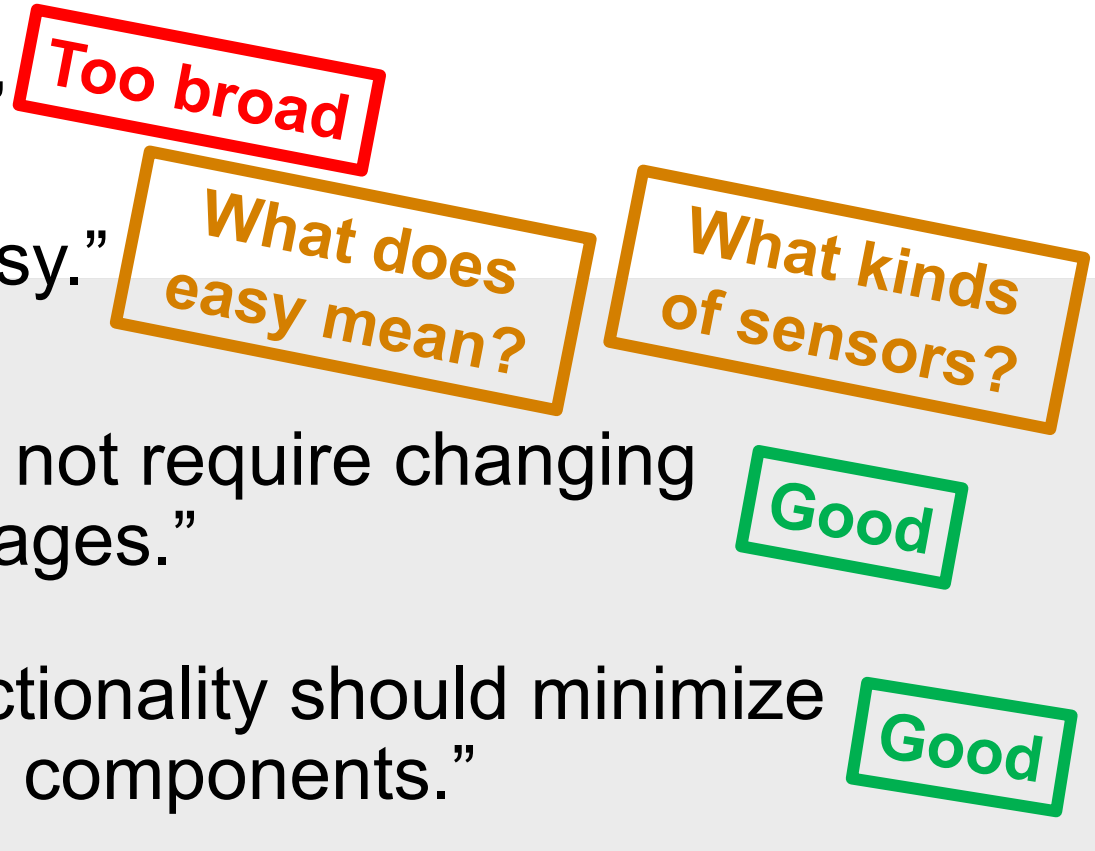


Less Extensible

Carnegie Mellon University

How to **Specify** Extensibility?

- "The system should be extensible."
- "Adding new sensors should be easy."
- "Adding new depth sensors should not require changing components that process depth images."
- "Adding new depth processing functionality should minimize changes to existing pre-processing components."



Carnegie Mellon University

Quality Attribute Specifications Should Be **Measurable** with a Concrete Metric

- **Hard Threshold** (e.g., **Performance**: “response time < 1s”, **Availability**: “99+% up time”, **Security**: “prevent attack”, …)
- **Soft Goal** (e.g., **Changeability**: “as little effort as possible”, **Performance**: “as fast as possible”, **Security**: “as little data compromised as possible”, …)

Not All Extensions Are The Same

- Publish-Subscribe easily supports:
  - Adding new **Publishers**
  - Adding new **Subscribers**
- Publish-Subscribe does not easily support:
  - Adding new **data fields** to the message types

Carnegie Mellon University

Quality Attribute Specifications Should Describe the **Scenario** that is being Measured

- What does the system respond to?
  - **Extensibility**: Which features are added?
  - **Performance**: The response time to which requests is measured?
  - **Robustness**: Which deviations from normal conditions are considered?
  - **Security**: What types of attacks should the system prevent?

Carnegie Mellon University

What Quality Attributes are Important for … ?

- Social Media Systems (e.g., Facebook)
- Mars Rovers
- Financial Trading Systems

Availability

Extensibility

Robustness

Performance

Again?!
Now with
Measurable Scenarios!

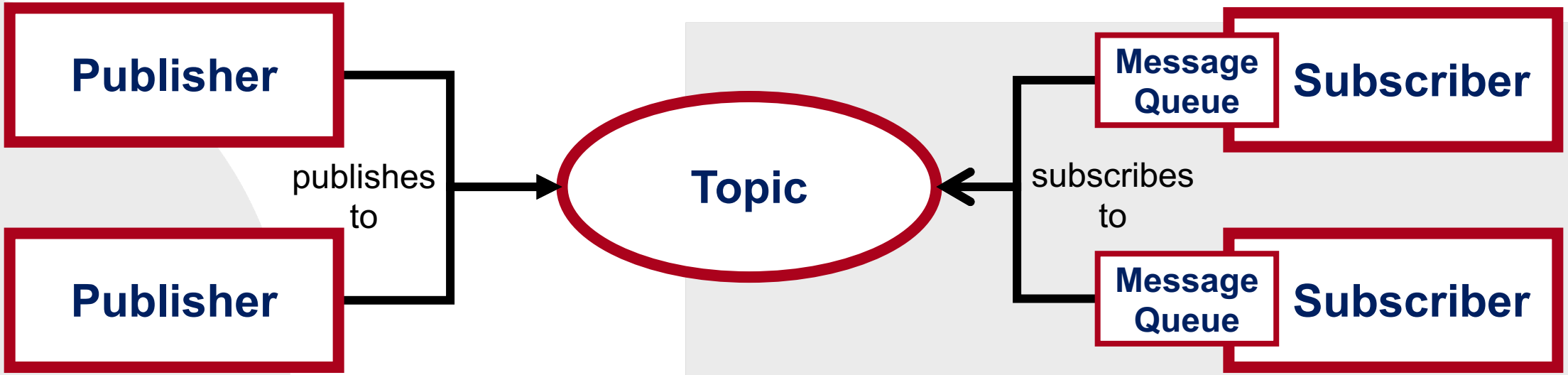
Carnegie Mellon University

String-based topic names are **error-prone** when topic names change in one place, but not another!
➔Lesson Learned: Communication Mechanisms can impact **Changeability**

What Could Go Wrong with ROS Topic Names?

```
pub = nh.advertise("topic_name");
pub.publish(msg)
```

```
nh.subscribe("topic_name", 10, callback);
```



Carnegie Mellon University

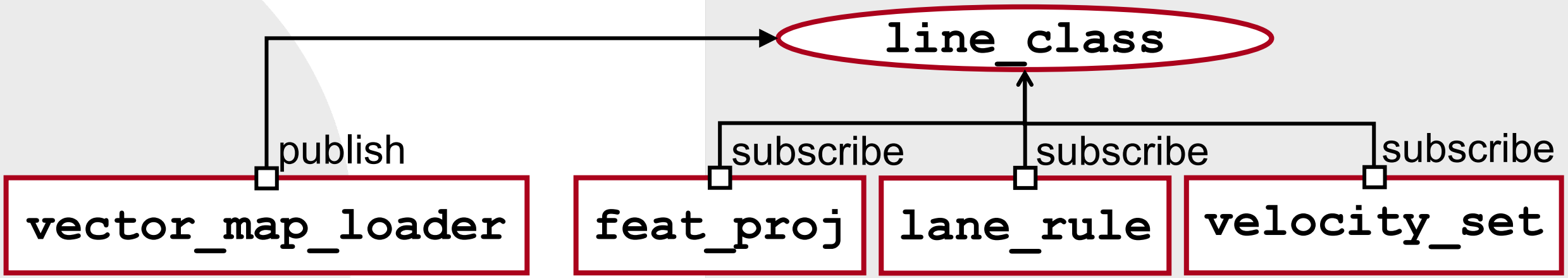
Most popular open-source framework for self-driving cars

Real-World Bug from Autoware.AI

Bug-introducing commit (inconsistent topic-renaming):

```
- ros::Publisher pub = n.advertise<[…]>("/line_class",[…]);
+ ros::Publisher pub = n.advertise<[…]>("/line",[…]);
```

Intended Architecture:



Carnegie Mellon University

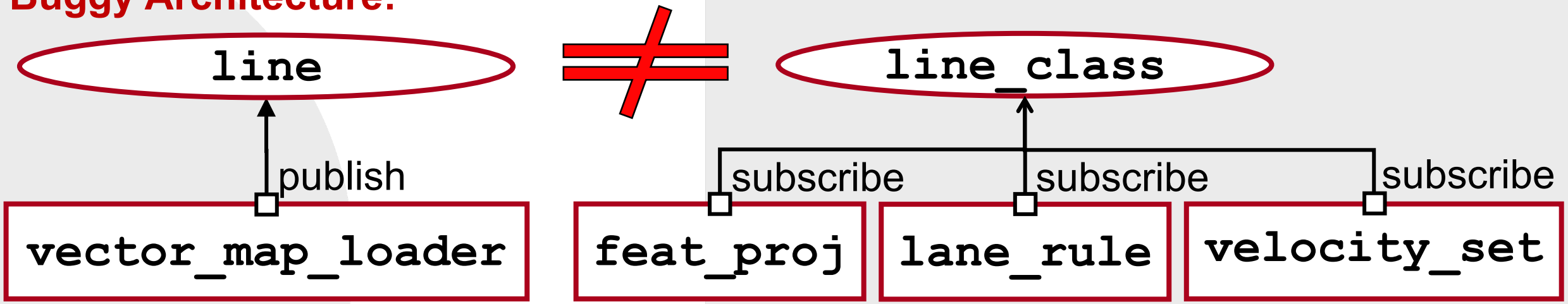
Lesson Learned: When using frameworks and libraries, be aware of their quality attributes, as even correct implementations can cause issues for your system!

Real-World Bug from Autoware.AI

Bug-introducing commit (inconsistent topic-renaming):

```
- ros::Publisher pub = n.advertise<[…]>("/line_class",[…]);
+ ros::Publisher pub = n.advertise<[…]>("/line",[…]);
```

Buggy Architecture:



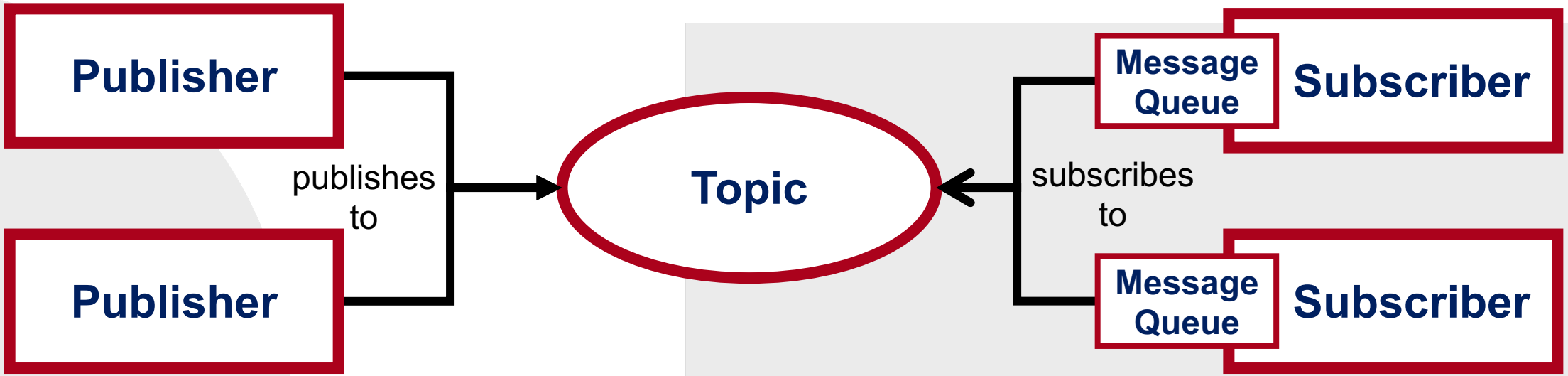
Carnegie Mellon University

Fixed queue sizes might lead to **message-loss** if subscribers cannot keep up with the publishing rate.
➔Lesson Learned: Functionality can depend on **Performance**

What Could Go Wrong with Message Queues?

```
pub = nh.advertise("topic_name");
pub.publish(msg)
```

```
nh.subscribe("topic_name", 10, callback);
```



Carnegie Mellon University

Lesson Learned: Some quality attributes (e.g., extensibility, changeability, testability) are **design-time concerns**, while others (e.g., performance, availability, scalability) are **run-time concerns**

Performance

- The degree to which a system minimizes the time between requests and responses (**response time**) or maximizes the number of responses within a given interval (**throughput**)
- In real-time systems **worse case response time** and **variance / jitter** are often more important
- Most client-server applications focus their requirements on predictability, so care more about **average case response time** and/or **throughput**
- High performance can positively impact **usability**

Carnegie Mellon University

Lesson learned: **Consider quality attributes early in the design process**

Limitations of ROS 1 motivated **Big Re-design** of the Framework towards ROS 2

- ROS 1 does not support **Real-Time Performance** requirements
  - No guarantees about message order or message delivery time
- ROS 1 does not support **Security** requirements
  - Any component can subscribe to any topic, listening to all messages
- **Effort** to update from ROS 1 to ROS 2 is quite **high**

Carnegie Mellon University

Lesson learned: **Consider quality attributes early in the design process**

Quality Attributes Are **Load-Bearing Walls**

- Quality attributes are very **hard to "add in later"**
- Design decisions that affect quality attributes are **hard to change**
- Early decisions in the architecture strongly impact the possible quality attributes of a system
- Often quality attributes are **cross-cutting** concerns, not localized in one part of the system, but spread throughout

Carnegie Mellon University

Lesson learned: In many cases quality attributes conflict with each other. This requires us to settle for a solution that is “good enough” for our goal.

Design Exercise

The average delay between sending data and receiving data is minimal

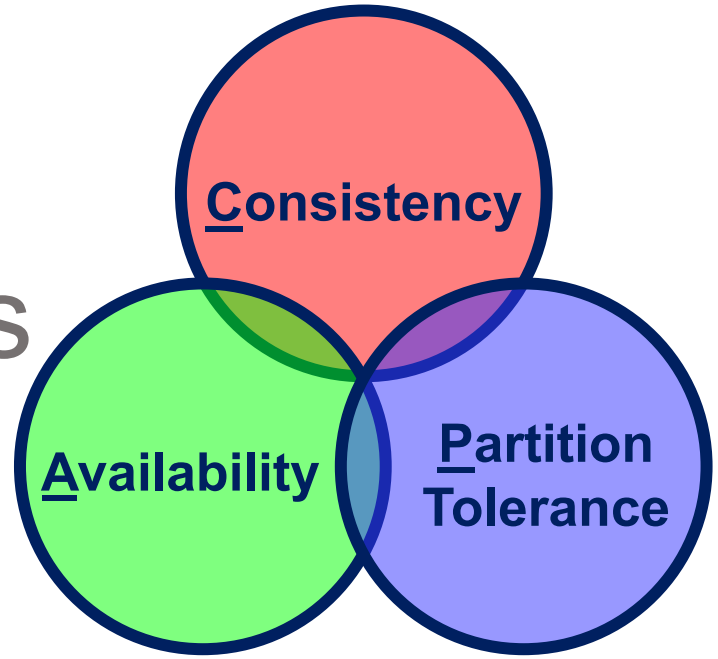
No messages are lost

How can we make publish-subscribe **fast**, **reliable**, and **secure**?

Data sent between components can only be read by authorized receivers

Carnegie Mellon University

Trade-offs in Database Systems



CAP Theorem: You can only pick 2 of:

- **Consistency** = Every read receives the most recent write or an error
- **Availability** = Every read receives a (non-error) response
- **Partition Tolerance** = The system continues to operate despite network failures

Carnegie Mellon University

Lesson learned: Since we **cannot** always **maximize all** quality attributes, we must **prioritize** which ones are most important and make **trade-offs** accordingly.

Examples of Quality Attribute Trade-offs

- **Security** vs. **Usability**
  - Two-factor authentications is **more secure** but **harder to use**
  - Remembering a long password is harder than
- **Security** vs. **Performance**
  - Encrypting and decrypting data **slows down** the system while making it **more secure**
- **Performance** vs. **Reliability**
  - TCP (**slow** but **reliable**) vs. UDP (**fast** but **unreliable**)

Carnegie Mellon University

Lesson learned: Whether quality attributes conflict with each other or support each other depends on the context and the design decision under consideration

Examples of Quality Attribute Synergies

- **Performance** and **Usability**
  - **Faster response times** make it **easier to use** interactive systems
- **Performance** and **Security**
  - **Faster intrusion detection** can keep the system more **secure**
- **Performance** and **Reliability**
  - Components with message queues **lose fewer messages** if they **process messages faster**
  - Highly **reliable connections** do not require many retries, resulting in **faster average case delivery**

Carnegie Mellon University

This is called **Attribute-Driven Design (ADD)**

How to Make Design Decisions in a Large Design Space?

Iteratively improve your design:

1. **Select** a quality attribute to improve (iteration goal)
2. **Chose** one or more parts of the system to refine
3. **Find & sketch** candidate solution & describe design decisions
4. **Analyze** candidate solution for iteration goal and other quality attributes
5. **Iterate** if necessary

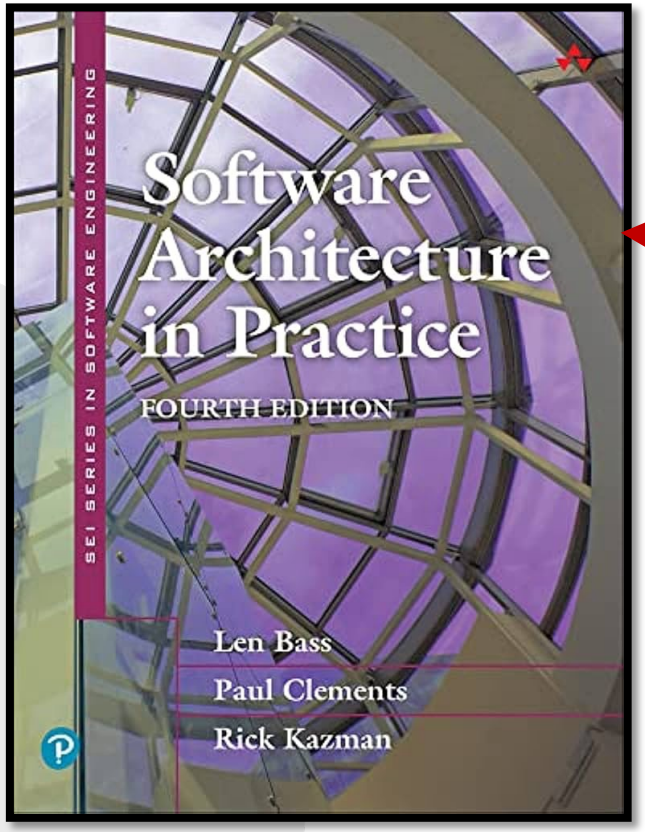
Carnegie Mellon University

Don’t Forget about the **Cost** of your Design!

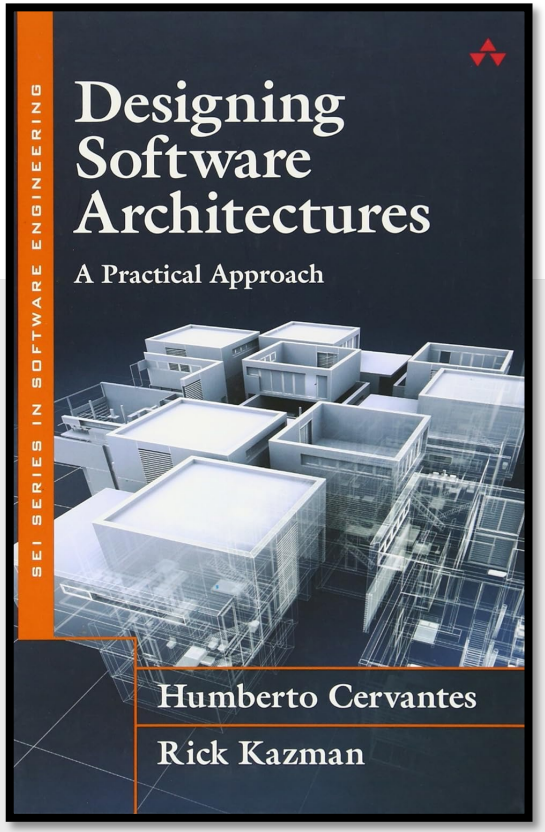
- Cost is not a quality attribute, but an important architectural driver that determines whether a design is feasible
- Cost can include the implementation effort, price of buying existing software, and operating cost

Carnegie Mellon University

Recommended Reading on Quality Attributes



Highly recommended for many topics throughout this course. Find it here (free with CMU account)



Carnegie Mellon University

Please Complete the Exit Ticket in Canvas!

Question 1 1 pts

Please briefly summarize the key messages from today's lecture (2~3 sentences).

Question 2 1 pts

Based on today's lecture, please describe **two quality attributes** that are important when designing a web-based appointment scheduling system for medical testing and describe whether they are **likely in conflict, in synergy, or independent** of each other.

Question 3 1 pts

Please leave any questions that you have about today's materials and things that are still unclear of confusing to you (if none, simply write N/A).

Carnegie Mellon University

Summary

- Functionality is not the only concern of software design
- Quality attributes measure the “goodness” of a design along a certain dimension
- Quality attribute specifications should be **measurable** and describe a **scenario** to which the system responds.
- Quality attributes are very hard to “add in later”, **so consider quality attributes early in the design process**
- Since we cannot always maximize all quality attributes, we must prioritize which ones are most important