Recitation 5: Architectural Modeling

17-445/645: Software Engineering for AI-Enabled Systems

Sept 27, 2019

Architectural Models

- Goal: Document & communicate design decisions & rationale
- Many existing notations & styles:
  - Informal (whiteboard)
  - Semi-formal (UML, SysML, C4)
  - Formal, mathematical models
- In this course:
  - We do not insist on a particular notation
  - BUT you should make sure your diagrams have:
    - Clear meanings with a key/legend
    - Consistent use of lines/boxes/...
    - Accompanying text if diagram is insufficient to convey info.

Signs of Bad Documentation

- Lines all look the same: Arrows don’t mean anything or could mean many things
- Inconsistent use of notations
- Lack of key or legend
- Too little or too much detail
- Implementation details mixed with architectural abstractions
- Missing relationships between parts
- Incomplete prose – poorly described designs
- No discussion of alternatives

Example

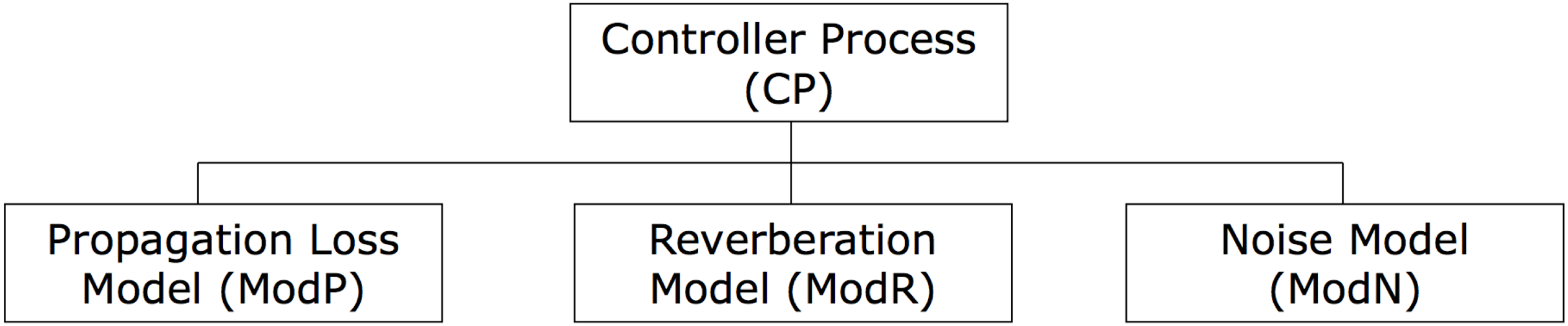
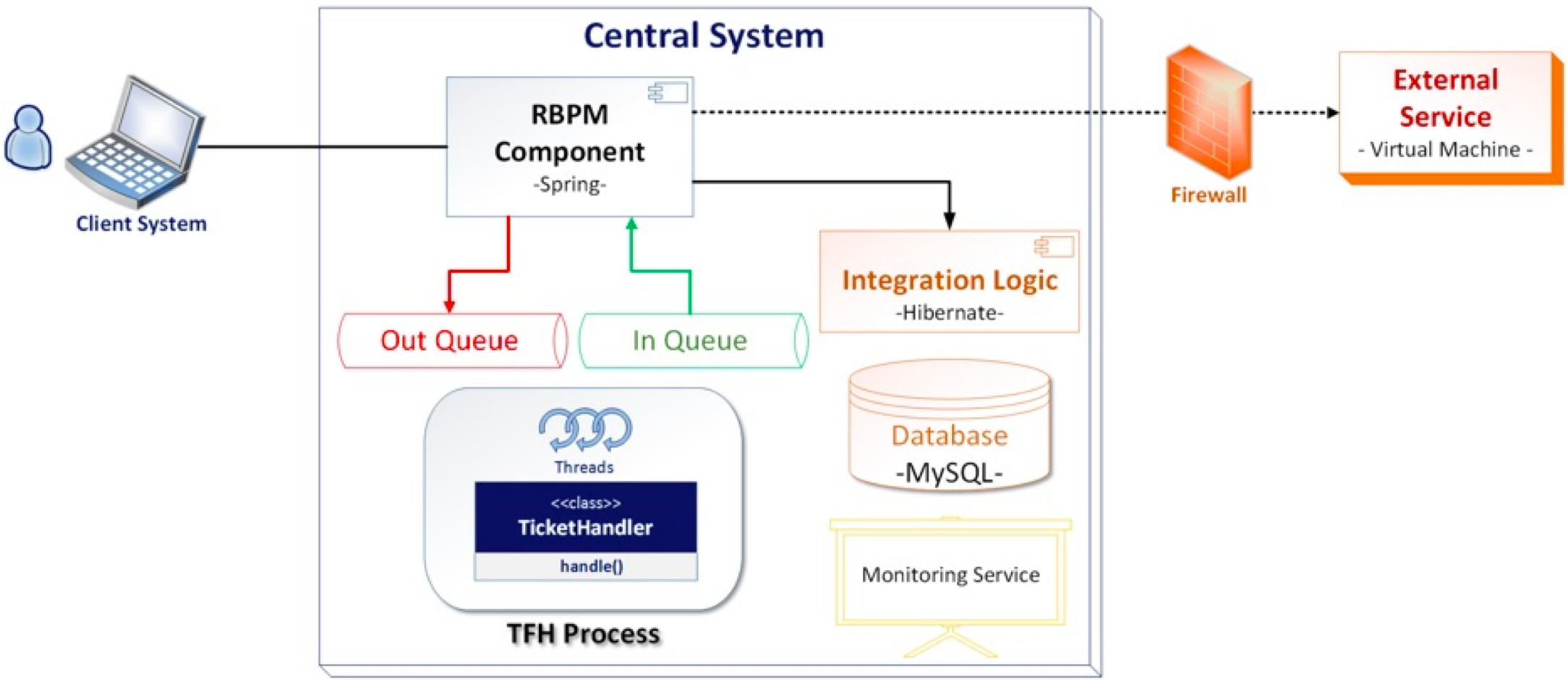


Figure X: Overall Software System Structure

Example


Central System
RBPM
Component
-Spring-
Client System
Out Queue
In Queue
Integration Logic
-Hibernate-
Threads
<<class>>
TicketHandler
handle()
TFH Process
Database
-MySQL-
Monitoring Service
Firewall
External
Service
- Virtual Machine -

A Methodology

1. Identify: What design decisions do I want to show?
2. Identify quality attributes (QAs) of interest
   - Focus on one or two related quality attributes in each diagram
3. Specify system entities
   - Use different shapes for different kinds of entities
4. If needed, annotate each entity with attributes
   - What information do I need to reason about QAs from Step 2?
5. Identify relationships between components
   - Use different types of edges for different relationships
   - Clearly define the meaning of an edge & its labels
6. Analyze the impact of the design decision(s) on QAs

Example: Google Glass Translation

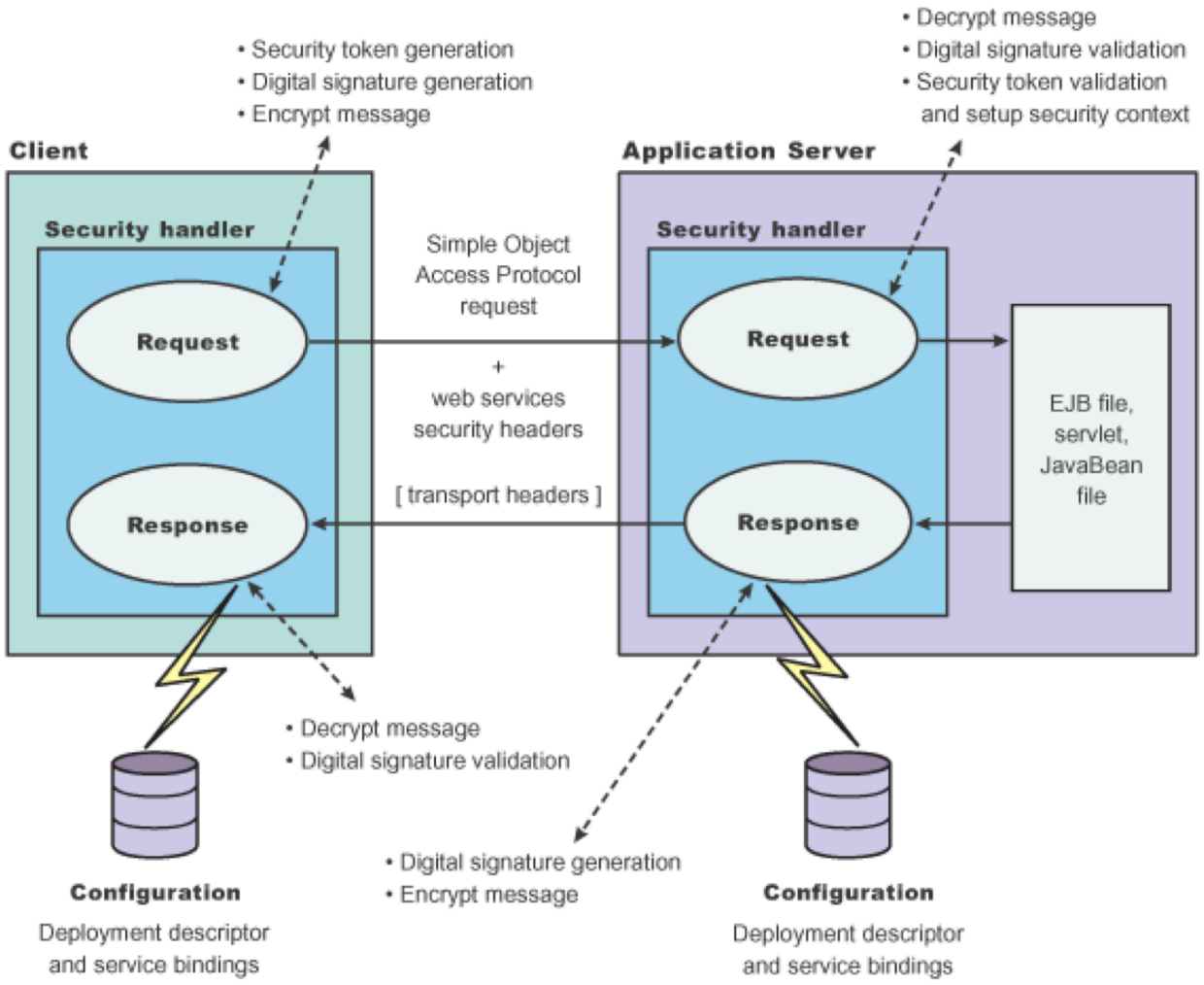
A Methodology

1. Identify: What design decisions do I want to show?
   - “Where should we perform ML tasks? Glass? Phone? Server?”
2. Identify quality attributes of interest
   - Latency (i.e., translation time), battery consumption
3. Specify system entities
   - ML tasks: Inference (OCR, language translation), model update
   - Platforms: Glass, phone, server
4. If needed, annotate each entity with attributes
   - ML tasks: Estimated amount of computation (no. operations, data size)
   - Platforms: Available computational resources (CPU power -- no. operations/sec, energy consumed/operation, available space, etc.)

A Methodology

5. Identify relationships between components
   - Edge between platforms: Data transfer, labeled with mb/sec
   - Edge from task to platform: Allocation of a task on a platform
6. Analyze the impact of the design decision(s) on quality attributes
   - What is the overall latency given the task allocation?
   - How much battery does the system consume?
   - Is this particular task allocation feasible?
   - What happens to the above two qualities if a connection between platforms is lost?
   - What happens when the model is updated?
   - etc.,

Another Example



- What are the design decisions being shown?
- What quality attributes are of importance?
- Is anything missing this diagram?

Key Takeaways

- Think about: What design decisions do I want to show in this diagram?
- Be consistent & unambiguous
- Don’t overload the meaning of lines/boxes/…
- Use multiple diagrams if needed; keep each simple
- Include text to supplement diagram if needed

It’s OK to invent your own notation, as long as it clearly & concisely convey the meaning!

References

- C4 model (http://c4model.com)
- UML (some subset of it may be useful)
  - Recommended book: UML Distilled by Fowler (CMU library access)
- ACME (https://www.cs.cmu.edu/~acme/index.html): A formal notation for architectural description