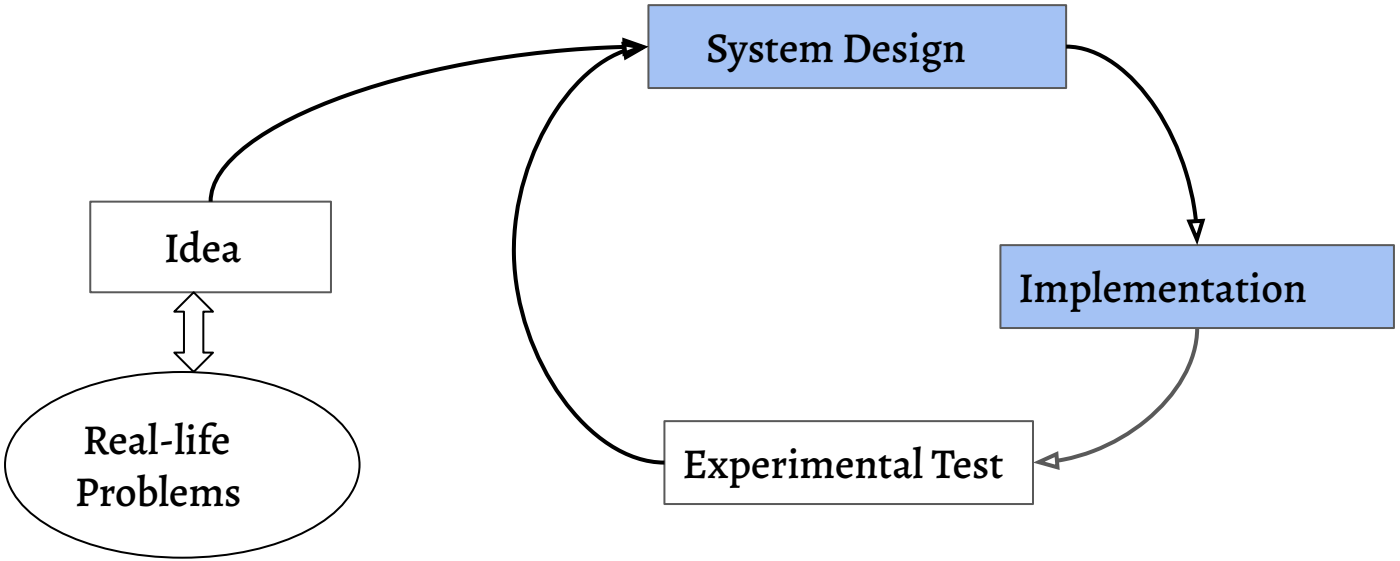
IIT KANPUR
Indian Institute of Technology Kanpur

CS888: Introduction to Profession and Communication

System Research: Design

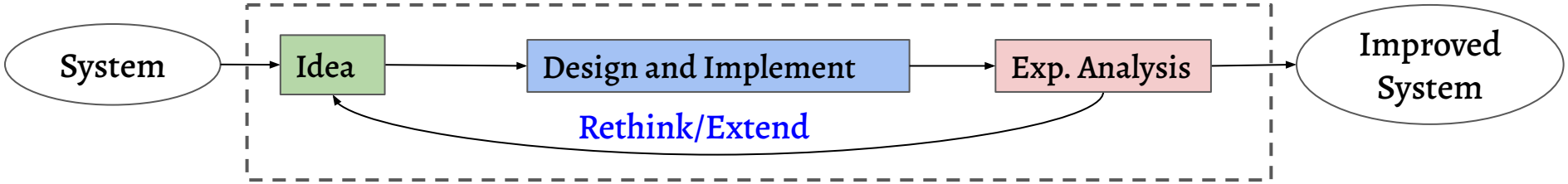
Debadatta Mishra, CSE, IIT Kanpur

Research involving “E” of CSE



- System Design: Efficiency, Security,
- Experiments → Analyze → {Validation of Idea or New design}

Research while engineering



- As mentioned previously, an idea or innovation is often preceded by analysis using experimental and other methods
- Implementation is engineering, but design is not!
- Insights ⟶ Improvement idea ⟶ Design ⟶ Implementation ⟶ Experiments

System design in CS research

- Objectives
  - What goals we want to achieve?
  - Which design elements are inefficient in the state of the art?

System design in CS research

- Objectives
  - What goals we want to achieve?
  - Which design elements are inefficient in the state of the art?
- Challenges
  - Non-trivial design aspects
  - An existing method can not be trivially applied

System design in CS research

- Objectives
  - What goals we want to achieve?
  - Which design elements are inefficient in the state of the art?
- Challenges
  - Non-trivial design aspects
  - An existing method can not be trivially applied
- Design tradeoffs
  - Most of the times we optimize one at the cost of other. Example: CPU vs. memory, security vs. efficiency etc.

Widely used system design principles

- Concurrency and parallelism
  - Simultaneous progress using multiple resources. Example: multi-threading, offloaded computing or data transfer
- Caching
- Pipelining
- Asynchronous/deferred processing
- Virtualization/indirections

Widely used system design principles

- Concurrency and parallelism
  - Simultaneous progress using multiple resources. Example: multi-threading, offloaded computing or data transfer
- Caching
  - Faster access leveraging the locality of reference properties, can give rise to consistency issues!

Widely used system design principles

- Concurrency and parallelism
  - Simultaneous progress using multiple resources. Example: multi-threading, offloaded computing or data transfer
- Caching
  - Faster access leveraging the locality of reference properties, can give rise to consistency issues!
- Pipelining
  - Breaking up the resources and computing into smaller elements; require careful coordination between stages

Widely used system design principles

- Asynchronous/deferred processing
  - Optimizing the critical path by delaying the processing
  - Typically a dedicated background “worker” handles the processing
  - Require careful coordination between the “worker” and the application
  - Example: Writing dirty disk blocks, Swapping

Widely used system design principles

- Asynchronous/deferred processing
  - Optimizing the critical path by delaying the processing
  - Typically a dedicated background “worker” handles the processing
  - Require careful coordination between the “worker” and the application
  - Example: Writing dirty disk blocks, Swapping
- Virtualization/indirections
  - Creation of a new layer between the application and resource
  - Require multiplexing by intercepting accesses
  - Should be careful about overheads!

Tools

- Code browsing
  - Directly examining large code-bases without having a high-level understanding is not advisable
  - Formulate the questions and try to answer them
  - Tools: cscope, eclipse, gdb

Tools

- Code browsing
  - Directly examining large code-bases without having a high-level understanding is not advisable
  - Formulate the questions and try to answer them
  - Tools: cscope, eclipse, gdb
- Coding best practices
  - Think modular, use comments, test at every significant milestones

Tools

- Code browsing
  - Directly examining large code-bases without having a high-level understanding is not advisable
  - Formulate the questions and try to answer them
  - Tools: cscope, eclipse, gdb
- Coding best practices
  - Think modular, use comments, test at every significant milestones
- Debugging
  - Use tools such as gdb, valgrind, simple print statements!