Motivation for Java

Concurrency & Parallelism

Douglas C. Schmidt
d.schmidt@vanderbilt.edu
www.dre.vanderbilt.edu/~schmidt

Professor of Computer Science
Institute for Software Integrated Systems
Vanderbilt University
Nashville, Tennessee, USA
Learning Objectives in this Lesson

- Understand the motivations for Java’s concurrent & parallel software support
Learning Objectives in this Lesson

- Understand the motivations for Java’s concurrent & parallel software support
- M1 – Leverage advances in hardware & software components
Learning Objectives in this Lesson

- Understand the motivations for Java’s concurrent & parallel software support
  - M1 – Leverage advances in hardware & software components
  - M2 – Improve software quality attributes
Learning Objectives in this Lesson

• Understand the motivations for Java’s concurrent & parallel software support
  • M1 – Leverage advances in hardware & software components
  • M2 – Improve software quality attributes
  • M3 – Support many popular services, apps, & capabilities
Motivation 1: Leveraging Advances in Commodity Hardware/Software
M1: Leveraging Advances in Commodity Hardware/Software

- Commodity hardware/software continues to improve

Commodity components are affordable & easy to obtain

See [www.dre.vanderbilt.edu/~schmidt/commodization.html](http://www.dre.vanderbilt.edu/~schmidt/commodization.html)
M1: Leveraging Advances in Commodity Hardware/Software

- Commodity hardware/software continues to improve, e.g.
  - Moore’s Law
  - # of transistors in an integrated circuit doubles every ~2 years

See en.wikipedia.org/wiki/Moore's_law
M1: Leveraging Advances in Commodity Hardware/Software

- Commodity hardware/software continues to improve, e.g.
  - Moore’s Law
  - Multi-core processors
    - Consist of 1 processor that contains 2+ CPUs that read & run instructions in parallel

See [en.wikipedia.org/wiki/Multi-core_processor](en.wikipedia.org/wiki/Multi-core_processor)
Commodity hardware/software continues to improve, e.g.:
- Moore’s Law
- Multi-core processors
  - Consist of 1 processor that contains 2+ CPUs that read & run instructions in parallel
  - Have become essential as clock speeds haven’t kept pace with transistor density

See [www.gotw.ca/publications/concurrency-ddj.htm](http://www.gotw.ca/publications/concurrency-ddj.htm)
M1: Leveraging Advances in Commodity Hardware/Software

- Commodity hardware/software continues to improve, e.g.
  - Moore’s Law
  - Multi-core processors
    - Consist of 1 processor that contains 2+ CPUs that read & run instructions in parallel
    - Have become essential as clock speeds haven’t kept pace with transistor density
      - It’s getting hard to buy a computer with only one core!

See postbiota.org/pipermail/tt/2012-February/010626.html
Commodity hardware/software continues to improve, e.g.:
- Moore’s Law
- Multi-core processors
  - Consist of 1 processor that contains 2+ CPUs that read & run instructions in parallel
  - Have become essential as clock speeds haven’t kept pace with transistor density
- Java 8 applies functional programming with fine-grained data parallelism to leverage proliferation of processor cores

See [www.infoq.com/presentations/parallel-java-se-8](http://www.infoq.com/presentations/parallel-java-se-8)
M1: Leveraging Advances in Commodity Hardware/Software

- Commodity hardware/software continues to improve, e.g.
  - Moore’s Law
  - Multi-core processors
  - Multi-threaded operating systems & virtual machines
  - Manage concurrent access to multi-core hardware & system resources

See en.wikipedia.org/wiki/{Operating_system, Virtual_machine}
M1: Leveraging Advances in Commodity Hardware/Software

- Commodity hardware/software continues to improve, e.g.
  - Moore’s Law
  - Multi-core processors
  - Multi-threaded operating systems & virtual machines
  - Multi-threaded middleware
    - Enhance productivity & performance via reusable app services

See [www.dre.vanderbilt.edu/~schmidt/concurrency-patterns.html](http://www.dre.vanderbilt.edu/~schmidt/concurrency-patterns.html)
M1: Leveraging Advances in Commodity Hardware/Software

- Commodity hardware/software continues to improve, e.g.
  - Moore’s Law
  - Multi-core processors
  - Multi-threaded operating systems & virtual machines
  - Multi-threaded middleware

Solid knowledge of concurrency & parallelism is needed to program multi-core systems effectively & efficiently
M2: Improving Software Quality Attributes
M2: Improving Software Quality Attributes

- Concurrency & parallelism help improve key software quality attributes

Software quality attributes are the overall factors that affect run-time behavior, system design, & experience.

See [www.sei.cmu.edu/reports/95tr021.pdf](http://www.sei.cmu.edu/reports/95tr021.pdf)
Concurrency & parallelism help improve key software quality attributes, e.g.

- Increase performance
- e.g., by leveraging multi-core processors & overlapping computing & communication
Concurrency & parallelism help improve key software quality attributes, e.g.

- Increase performance
- Improve responsiveness
  - e.g., don’t ignore user input while other processing is occurring in background threads
Concurrency & parallelism help improve key software quality attributes, e.g.

- Increase performance
- Improve responsiveness
- Simplify program structure
  - e.g., by using synchronous calls that are easier to understand & sustain over the lifecycle
M2: Improving Software Quality Attributes

• Concurrency & parallelism help improve key software quality attributes, e.g.
  • Increase performance
  • Improve responsiveness
  • Simplify program structure

See the next lesson on “Benefits of Concurrency in Java & Android” for more discussion
M3: Supporting Popular Services, Apps, & Capabilities
M3: Supporting Popular Services, Apps, & Capabilities

- Concurrency & parallelism is important in various contexts
Concurrency & parallelism is important in various contexts, e.g.
- Scalable heterogeneous multi-core processing for performance-intensive apps

See www.androidauthority.com/why-8-and-10-cpu-cores-in-smartphones-are-a-good-idea-607894
Concurrency & parallelism is important in various contexts, e.g.
- Scalable heterogeneous multi-core processing for performance-intensive apps, e.g.
- Mobile gaming & image recognition

See [www.nvidia.com/content/PDF/tegra_white_papers/tegra-whitepaper-0911a.pdf](http://www.nvidia.com/content/PDF/tegra_white_papers/tegra-whitepaper-0911a.pdf)
Concurrency & parallelism is important in various contexts, e.g.

- Scalable heterogeneous multi-core processing for performance-intensive apps, e.g.
  - Mobile gaming & image recognition
  - Web servers that support e-commerce & social media

See [www.dre.vanderbilt.edu/JAWS](http://www.dre.vanderbilt.edu/JAWS)
Concurrency & parallelism is important in various contexts, e.g.

- Scalable heterogeneous multi-core processing for performance-intensive apps, e.g.
  - Mobile gaming & image recognition
  - Web servers that support ecommerce & social media
  - "Big data" analytics

See [www.oracle.com/technetwork/articles/java/fork-join-422606.html](http://www.oracle.com/technetwork/articles/java/fork-join-422606.html)
Concurrency & parallelism is important in various contexts, e.g.
- Scalable heterogeneous multi-core processing for performance-intensive apps
- Responsive user interactions, e.g.
  - Let user do productive work while the system is busy
  - Give intermediate results, before operation is finished
  - If waiting is inevitable provide a progress indicator

See [en.wikipedia.org/wiki/Responsiveness#Solutions_to_Improve_Responsiveness](en.wikipedia.org/wiki/Responsiveness#Solutions_to_Improve_Responsiveness)
Concurrency & parallelism is important in various contexts, e.g.
- Scalable heterogeneous multi-core processing for performance-intensive apps
- Responsive user interactions
- Sustainable software lifecycles, e.g.
  - Resilient changing requirements & environments

See www.dre.vanderbilt.edu/POSA
End of Motivation for Java
Concurrency & Parallelism