Yu Chen
October 8, 2014

Research Document

Dual State Framework - Software Development Project 2014 - IT Carlow

Introduction

The aim of this document is to outline the research conducted to obtain a greater understanding about the technologies that the **Dual State Framework** will use for the development.

The project is going to create a framework which implements parallel computing. It helps game developers be easier to parallelize. The research contains Game Development, Parallel Programming Model, Concurrency Model, and etc.

Dual State Framework will be developed by C++ under Mac Os X. OpenMP or Intel TBB is used for implementing Parallel Computing. In addition, it will be compiled and debugged both on Microsoft Windows and Linux as well. This process will be done by Cmake.

The API Documentation of this project will be created by Doxygen.

A simple game will be written for testing this library. This game will be developed by C++ with SFML or SDL library.

Git will be used for source control and version control. The project is now available in following link.

https://github.com/kuyoonjo/DualStateFramework

Game Development

Games development is one of the most exciting areas of software development that you can work in. Game developers require skills and expertise in areas such as software development (software design and programming), game design, graphics programming, modeling, simulation and animation.

Graphics APIs

*"**OpenGL** is the premier environment for developing portable, interactive 2D and 3D graphics applications. Since its introduction in 1992, OpenGL has become the industry's most widely used and supported 2D and 3D graphics application programming interface (API), bringing thousands of applications to a wide variety of computer platforms. OpenGL fosters innovation and speeds application development by incorporating a broad set of rendering, texture mapping, special effects, and other powerful visualization functions. Developers can leverage the power of OpenGL across all popular desktop and workstation platforms, ensuring wide application deployment. "* [OpenGL OverviewData Encryption]

*"**Microsoft DirectX** is the graphics technology powering today's most impressive games. The latest version, DirectX 11, enables the addition of advanced effects and features in NVIDIA-enhanced titles, ranging from tessellation and HBAO+, to Percentage Closer Soft Shadows and NVIDA HairWorks. "* [DirectX 11]

Multimedia Libraries

*"**SFML** (Simple and Fast Multimedia Library) is a portable API for multimedia programming. It is written in C++ with bindings available for C, D, Python, Ruby, OCaml, .Net and Go. It can be thought of as an object oriented alternative to SDL. SFML provides hardware accelerated 2D graphics using OpenGL, supports OpenGL windowing and provides different modules that ease multimedia and game programming. SFML site offers complete SDK bundle in single pack, and tutorials to ease the developers."* [SFML]

*"**SDL** (Simple DirectMedia Layer) is a cross-platform development library designed to provide low level access to audio, keyboard, mouse, joystick, and graphics hardware via OpenGL and Direct3D. It is used by video playback software, emulators, and popular games including Valve's award winning catalog and many Humble Bundle games."* [SDL]

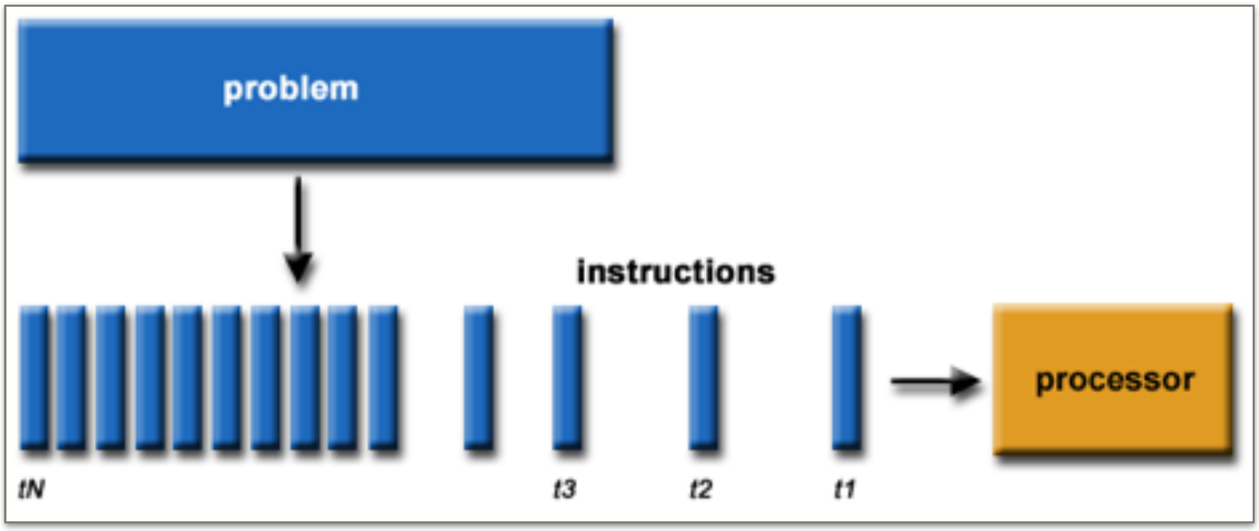
Parallel Programming Model

Parallel programming model is a set of software technologies to express parallel algorithms and match applications with the underlying parallel systems.

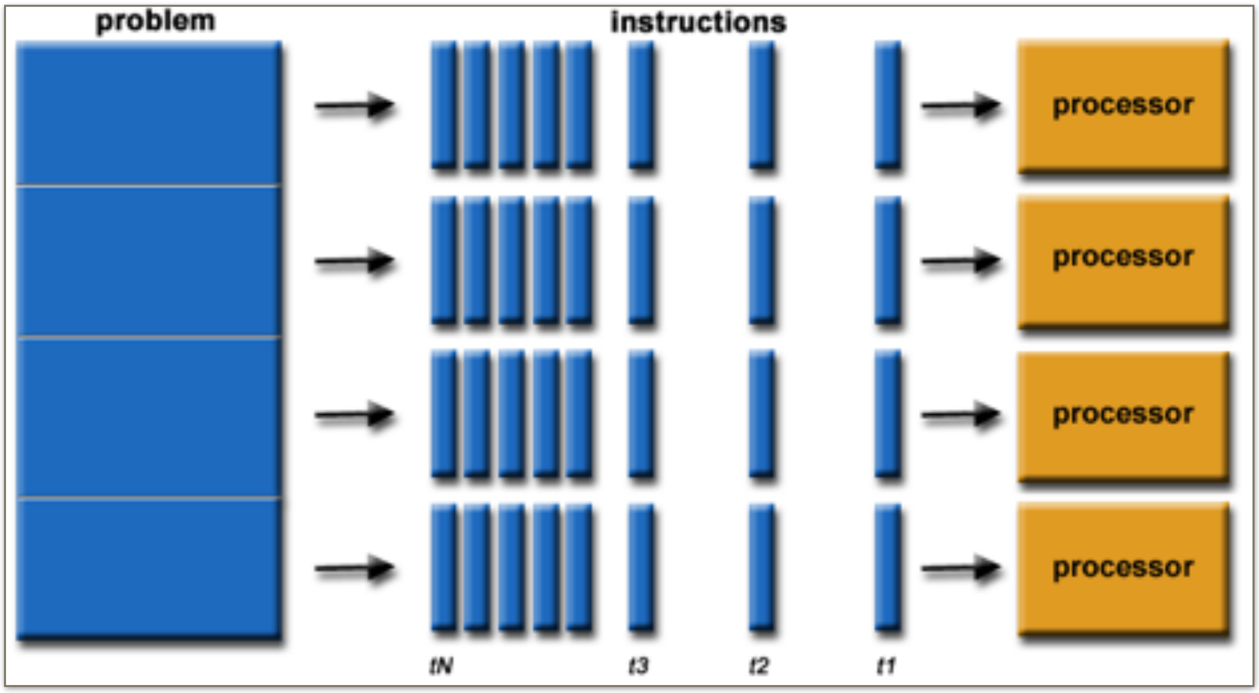
Serial Computing vs Parallel Computing

Serial Computing	**Parallel Computing**
A problem is broken into a discrete series of instructions	A problem is broken into discrete parts that can be solved concurrently
Instructions are executed sequentially one after another	Each part is further broken down to a series of instructions
Executed on a single processor	Instructions from each part execute simultaneously on different processors
Only one instruction may execute at any moment in time	An overall control/coordination mechanism is employed

Serial Computing Diagram: [Introduction to Parallel Computing]

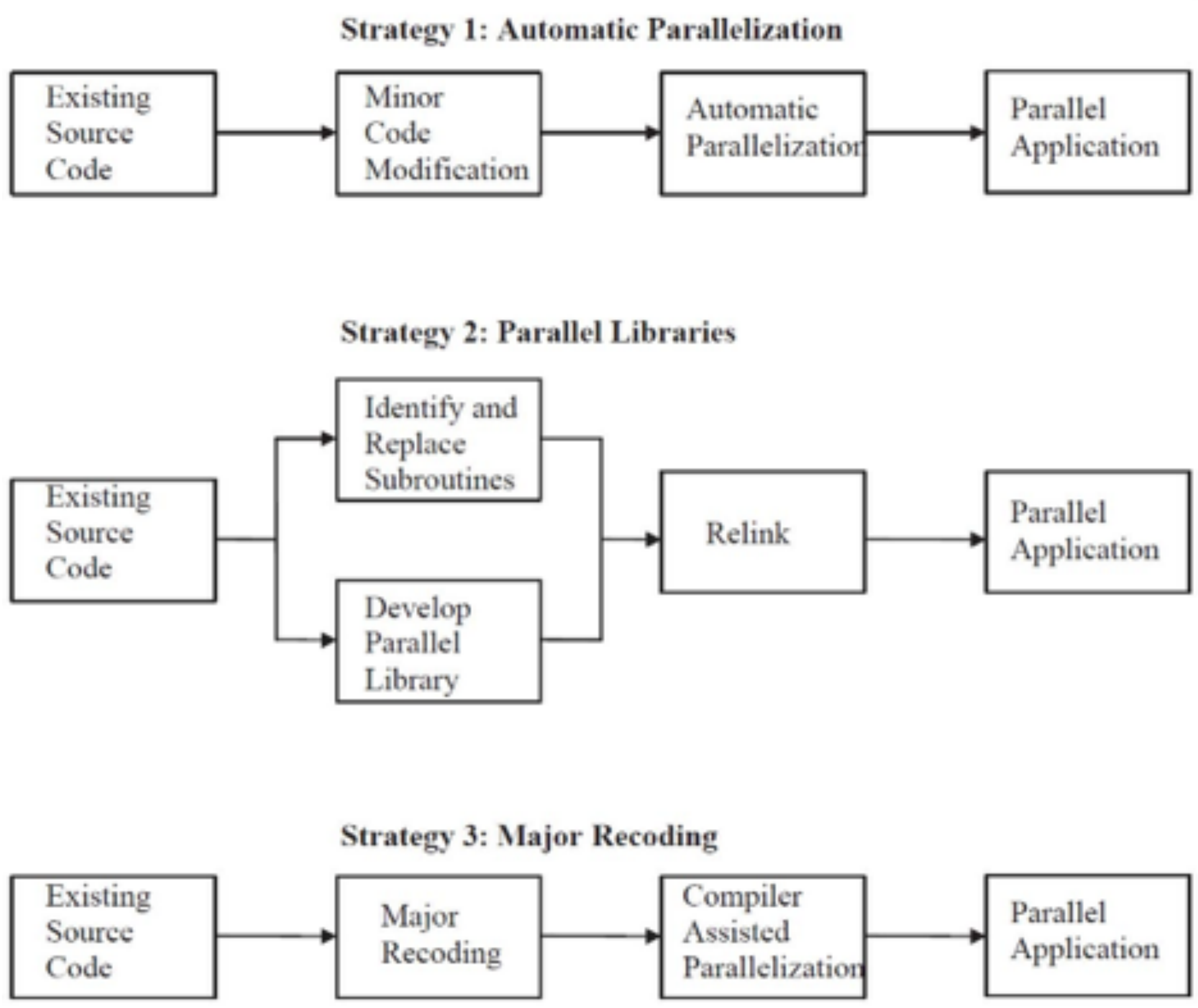


Parallel Computing Diagram: [Introduction to Parallel Computing]



Developing Parallel Programs

Parallel Programming Process Diagrams: [Parallel Programming Models]



POSIX Threads

"In shared memory multiprocessor architectures, threads can be used to implement parallelism. Historically, hardware vendors have implemented their own proprietary versions of threads, making portability a concern for software developers. For UNIX systems, a standardized C language threads programming interface has been specified by the IEEE POSIX 1003.1c standard. Implementations that adhere to this standard are referred to as POSIX threads, or Pthreads." [POSIX Threads Programming]

OpenMP

"The OpenMP (Open Multi-Processing) specification is a standard for a set of compiler directives, library routines, and environment variables that can be used to specify shared memory parallelism in Fortran and C/C++ programs." [OpenMP®/Clang]

"This specification provides a model for parallel programming that is portable across shared memory architectures from different vendors. Compilers from numerous vendors support the OpenMP API. " [OpenMP Application Program Interface]

More information about the OpenMP API can be found at http://www.openmp.org .

Intel TBB

"Intel® Threading Building Blocks (Intel® TBB) 4.3 is a widely used, award-winning C and C++ library for creating high performance, scalable parallel applications.

- *Enhance Productivity and Reliability - Rich set of components to efficiently implement higher-level, task-based parallelism*
- *Gain Performance Advantage- Future-proof applications to tap multicore and many-core processing power*
- *Portability and Compatibility –Open source and commercial licensing. Supports Windows*, Linux*, OS X*, and Android* (additional with open source). It's compatible with multiple compilers and Intel compatible processors including Intel® Atom™, Core™, Xeon® processors, and Intel® Xeon Phi™ coprocessors."* [Intel® Threading Building Blocks]

Concurrency Model

Concurrency is a property of systems in which several computations are executing simultaneously, and potentially interacting with each other.

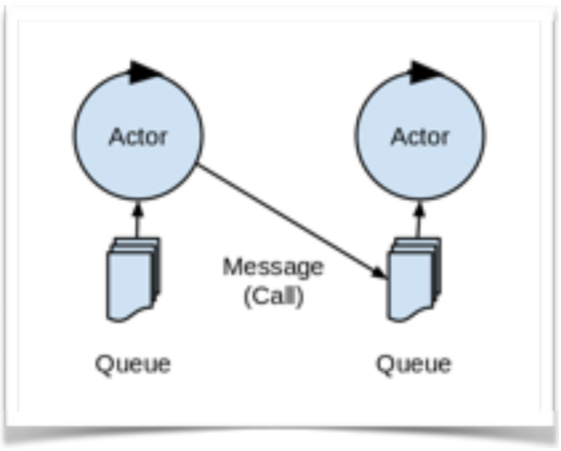
"The term Concurrency refers to techniques that make program more usable. Concurrency can be implemented and is used a lot on single processing units, nonetheless it may benefit from multiple processing units with respect to speed. If an operating system is called a multi-tasking operating system, this is a synonym for supporting concurrency. If you can load multiple documents simultaneously in the tabs of your browser and you can still open menus and perform more actions, this is concurrency." [Parallelism vs. Concurrency]Concurrency vs Parallelism

Concurrency	Parallelism
Concurrency is when two tasks can start, run, and complete in overlapping time periods. It doesn't necessarily mean they'll ever both be running at the same instant. Eg. multitasking on a single-core machine.	**Parallelism** is when tasks literally run at the same time. Eg. on a multicore processor.
Eg. multitasking on a single-core machine.	Eg. on a multicore processor.
v A condition that exists when at least two threads are making progress. A more generalized form of parallelism that can include time-slicing as a form of virtual parallelism.	A condition that arises when at least two threads are executing simultaneously.

Actors Model

"A general model of concurrent computation developed by CarlHewitt, HenryBaker and GulAgha (also "actor model"). Several ActorLanguages are based on this model. Actors are autonomous and concurrent objects which execute asynchronously. The actors model provides flexible mechanisms for building parallel and distributed software systems. " [Actors Model]

"An Actor is like an object instance executed by a single thread. Instead of direct calls to methods, messages are put into the Actors "mailbox" (~queue). The actor single threaded reads and processes messages from the queue sequentially (with exceptions). Internal state is exposed/shared by passing messages (with copied state) to other Actors." [Comparison of different concurrency models]



CSP (Communicating Sequential Processes)

"Communicating Sequential Processes, or CSP, is a language for describing patterns of interaction. It is supported by an elegant, mathematical theory, a set of proof tools, and an extensive literature. The book Communicating Sequential Processes was first published in 1985 by Prentice Hall International (who have kindly released the copyright); it is an excellent introduction to the language, and also to the mathematical theory." [Communicating Sequential Processes (CSP)]

"CSP systems can be seen as a special actor system having bounded mailboxes (queues) of size 0. So if one process (~Actor) wants to pass a message to another process, the caller is blocked until the receiver accepts the message. Alternatively to being blocked, a CSP-process can choose to do other things e.g. check for incoming messages (this introduces non determinism to the order of outgoing messages). A receiver cannot accept incoming

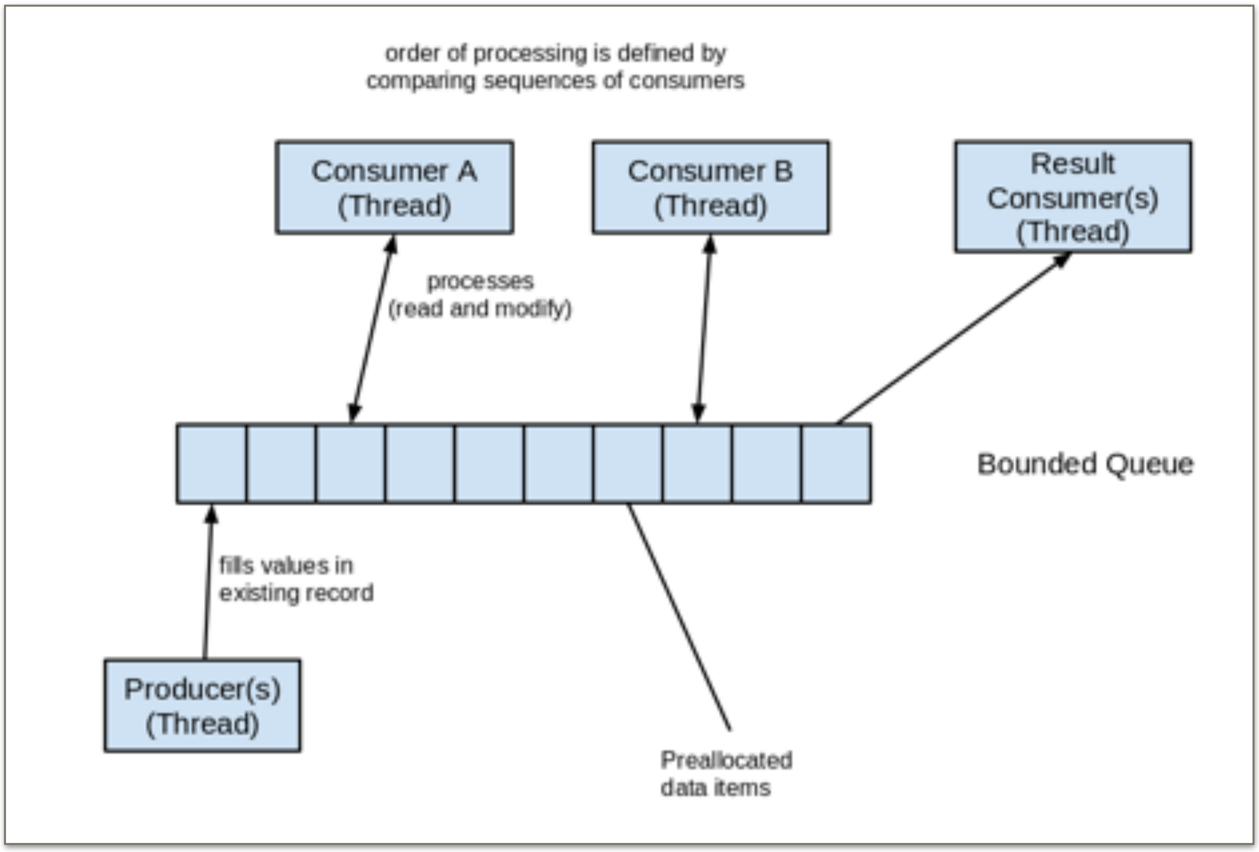
messages if he is occupied with processing a prior one." [Comparison of different concurrency models]

Disruptor

The disruptor data structure came up some years ago and was invented and pioneered by Martin Thompson, Mike Barker, and Dave Farley at LMAX exchange.

git: https://github.com/LMAX-Exchange/disruptor

"The disruptor is a bounded queue (implemented by a ring buffer) where producers add to the head of the queue (if slots are available, else the producer is blocked). Consumers access the queue at different points, so each consumer has its own read position (cursor) inside the queue. Sequencing is used to manage queue access and processing order." [Comparison of different concurrency models]


order of processing is defined by comparing sequences of consumers
Consumer A (Thread)
Consumer B (Thread)
Result Consumer(s) (Thread)
processes (read and modify)
Bounded Queue
fills values in existing record
Producer(s) (Thread)
Preallocated data items

Thread

A thread is a basic unit of CPU utilization, consisting of a program counter, a stack, and a set of registers, (and a thread ID.) Traditional (heavyweight) processes have a single thread of control - There is one program counter, and one sequence of instructions that can be carried out at any given time.

Some operating system provide a combined user level thread and Kernel level thread facility. Solaris is a good example of this combined approach. In a combined system, multiple threads within the same application can run in parallel on multiple processors and a blocking system call need not block the entire process. Multithreading models are three types

- Many to many relationship.
- Many to one relationship.
- One to one relationship.

Development Tools

IDEs

- *"**Xcode** provides everything developers need to create great applications for Mac, iPhone, and iPad. Xcode brings user interface design, coding, testing, and debugging all into a unified workflow. The Xcode IDE combined with the Cocoa and Cocoa Touch frameworks, and the Swift programming language make developing apps easier and more fun than ever before. Xcode includes the Xcode IDE, Swift and Objective-C compilers, Instruments analysis tool, iOS Simulator, the latest OS X and iOS SDKs, and hundreds of powerful features."* [Xcode]

- *"**NetBeans** IDE lets you quickly and easily develop Java desktop, mobile, and web applications, as well as HTML5 applications with HTML, JavaScript, and CSS. The IDE also provides a great set of tools for PHP and C/C++ developers. It is free and open source and has a large community of users and developers around the world."* [NetBeans IDE Features]

- *"**Eclipses CDT** provides a fully functional C and C++ Integrated Development Environment based on the Eclipse platform. Features include: support for project creation and managed build for various toolchains, standard make build, source navigation, various source knowledge tools, such as type hierarchy, call graph, include browser, macro definition browser, code editor with syntax highlighting, folding and hyperlink navigation, source code refactoring and code generation, visual debugging tools, including memory, registers, and disassembly viewers."* [Eclipse CDT (C/C++ Development Tooling)]

Documentation

*"**Doxygen** is the de facto standard tool for generating documentation from annotated C++ sources, but it also supports other popular programming languages such as C, Objective-C, C#, PHP, Java, Python, IDL (Corba, Microsoft, and UNO/OpenOffice flavors), Fortran, VHDL, Tcl, and to some extent D.*

Doxygen can help you in three ways:

1. *It can generate an on-line documentation browser (in HTML) and/or an off-line reference manual (in Latex) from a set of documented source files. There is also support for generating output in RTF (MS-Word), PostScript, hyperlinked PDF, compressed HTML, and Unix man pages. The documentation is extracted directly from the sources, which makes it much easier to keep the documentation consistent with the source code.*

2. *You can configure doxygen to extract the code structure from undocumented source files. This is very useful to quickly find your way in large source distributions. Doxygen can also visualize the relations between the various elements by means of include dependency graphs, inheritance diagrams, and collaboration diagrams, which are all generated automatically.*
3. *You can also use doxygen for creating normal documentation (as I did for the doxygen user manual and web-site).*

Doxygen is developed under Mac OS X and Linux, but is set-up to be highly portable. As a result, it runs on most other Unix flavors as well. Furthermore, executables for Windows are available." [Doxygen]

Others

*"**CMake** is an extensible, open-source system that manages the build process in an operating system and in a compiler-independent manner. Unlike many cross-platform systems, CMake is designed to be used in conjunction with the native build environment. Simple configuration files placed in each source directory (called CMakeLists.txt files) are used to generate standard build files (e.g., makefiles on Unix and projects/workspaces in Windows MSVC) which are used in the usual way. CMake can generate a native build environment that will compile source code, create libraries, generate wrappers and build executables in arbitrary combinations. CMake supports in-place and out-of-place builds, and can therefore support multiple builds from a single source tree. CMake also supports static and dynamic library builds. Another nice feature of CMake is that it generates a cache file that is designed to be used with a graphical editor. For example, when CMake runs, it locates include files, libraries, and executables, and may encounter optional build directives. This information is gathered into the cache, which may be changed by the user prior to the generation of the native build files."* [About CMake]

Reference

https://www.opengl.org/about/ [OpenGL OverviewData Encryption]

http://www.geforce.com/hardware/technology/dx11 [DirectX 11]

http://www.sfml-dev.org/index.php [SFML]

https://computing.llnl.gov/tutorials/parallel_comp/ [Introduction to Parallel Computing]

http://ecar2012.hpclatam.org/files/Foundations_OpenMP_Clase2.pdf [Parallel Programming Models]

https://computing.llnl.gov/tutorials/pthreads/ [POSIX Threads Programming]

http://clang-omp.github.io [OpenMP®/Clang]

http://www.openmp.org/mp-documents/OpenMP4.0.0.pdf [OpenMP Application Program Interface]

https://software.intel.com/en-us/intel-tbb [Intel® Threading Building Blocks]

http://www.haskell.org/haskellwiki/Parallelism_vs._Concurrency [Parallelism vs. Concurrency]

http://c2.com/cgi/wiki?ActorsModel [Actors Model]

http://java-is-the-new-c.blogspot.ie/2014/01/comparision-of-different-concurrency.html [Comparison of different concurrency models]

http://www.usingcsp.com [Communicating Sequential Processes (CSP)]

https://itunes.apple.com/us/app/xcode/id497799835 [Xcode]

https://netbeans.org/features/index.html [NetBeans IDE Features]

http://www.eclipse.org/cdt/ [Eclipse CDT (C/C++ Development Tooling)]

http://www.stack.nl/~dimitri/doxygen/ [Doxygen]

http://www.cmake.org/overview/ [About CMake]