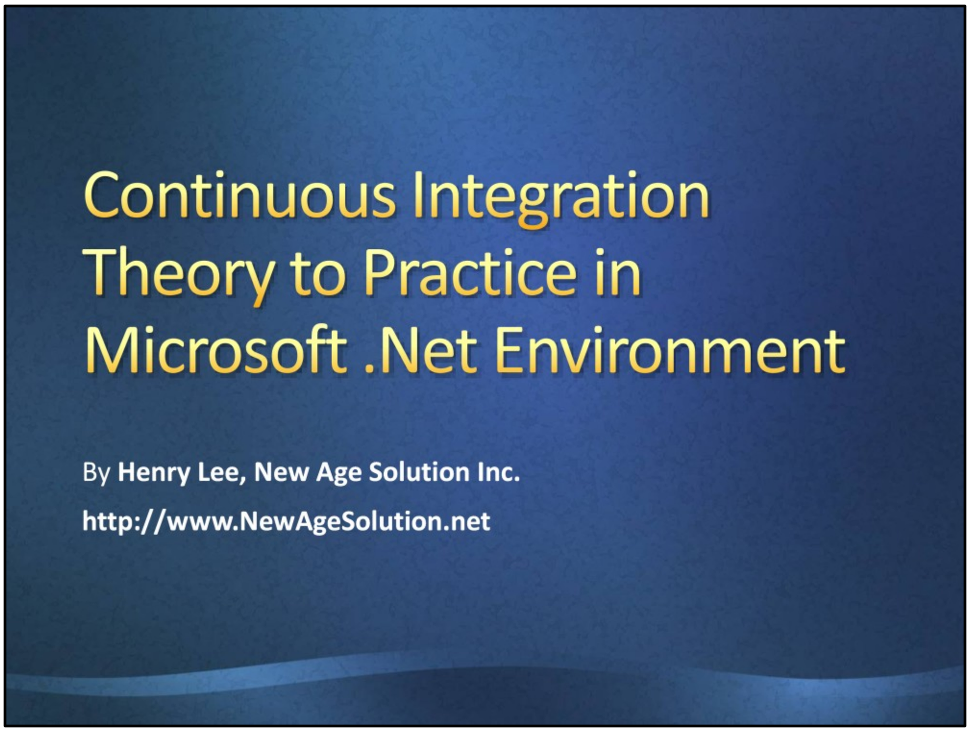
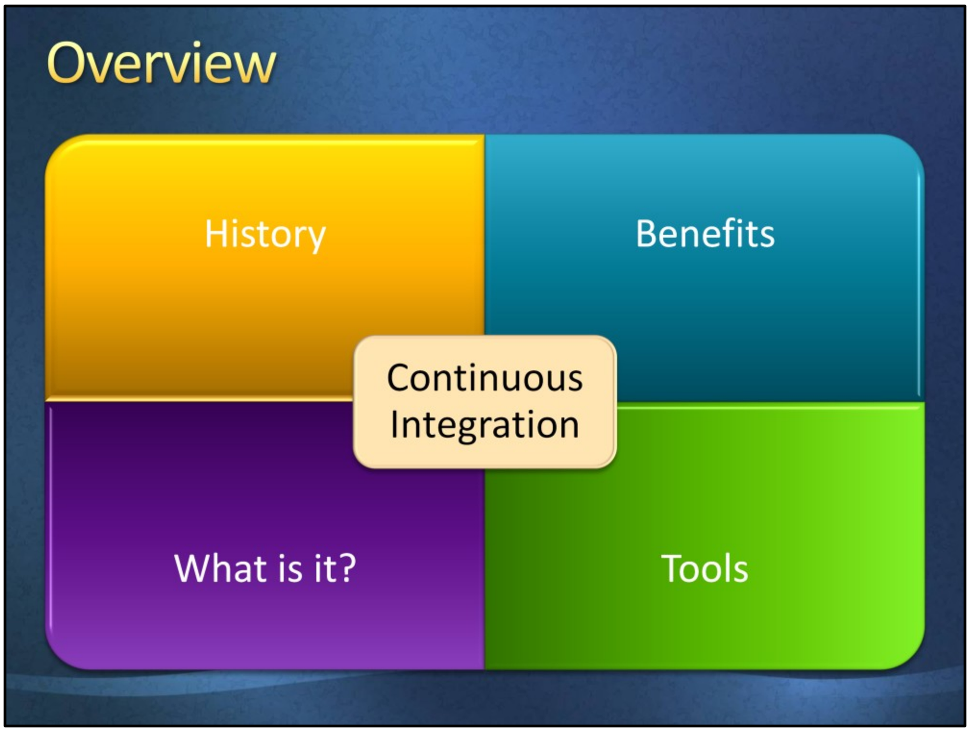


Continuous Integration Theory to Practice in Microsoft .Net Environment

By **Henry Lee, New Age Solution Inc.**

http://www.NewAgeSolution.net

We (http://www.NewAgeSolution.net) have extensive experience in enterprise and system architectures, system engineering, project management, and software design and development. We will be presenting a software development practice known as "Continuous Integration" and how Continuous Integration can help improve and simplify complex software development process. We will also cover how Continuous Integration can improve software quality and reduce risk.

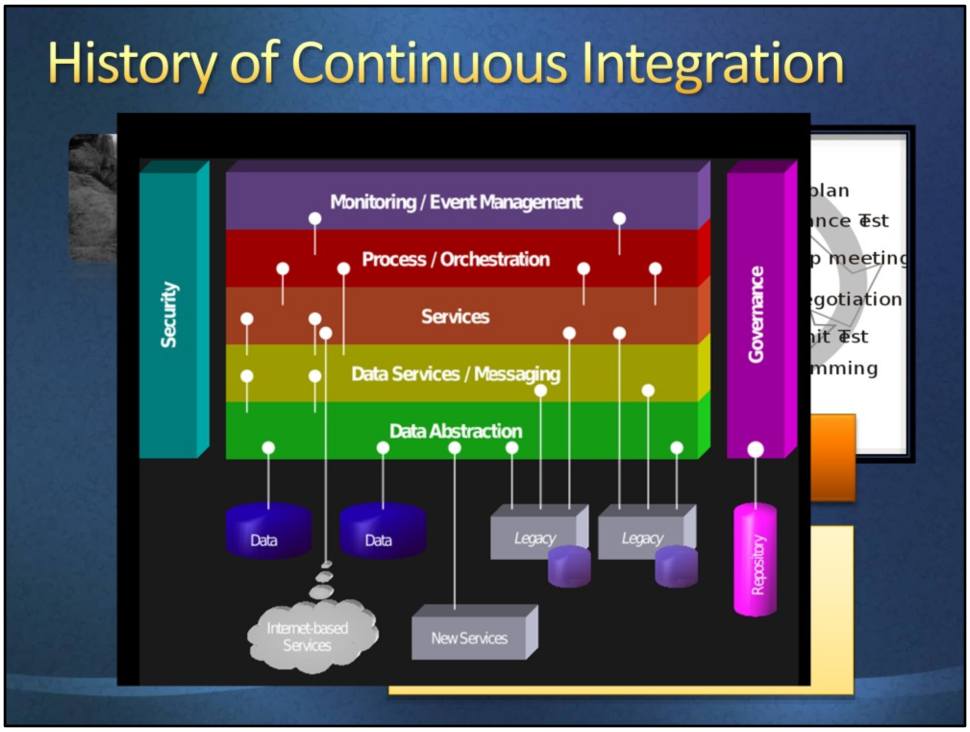


Overview

History
Benefits
Continuous Integration
What is it?
Tools

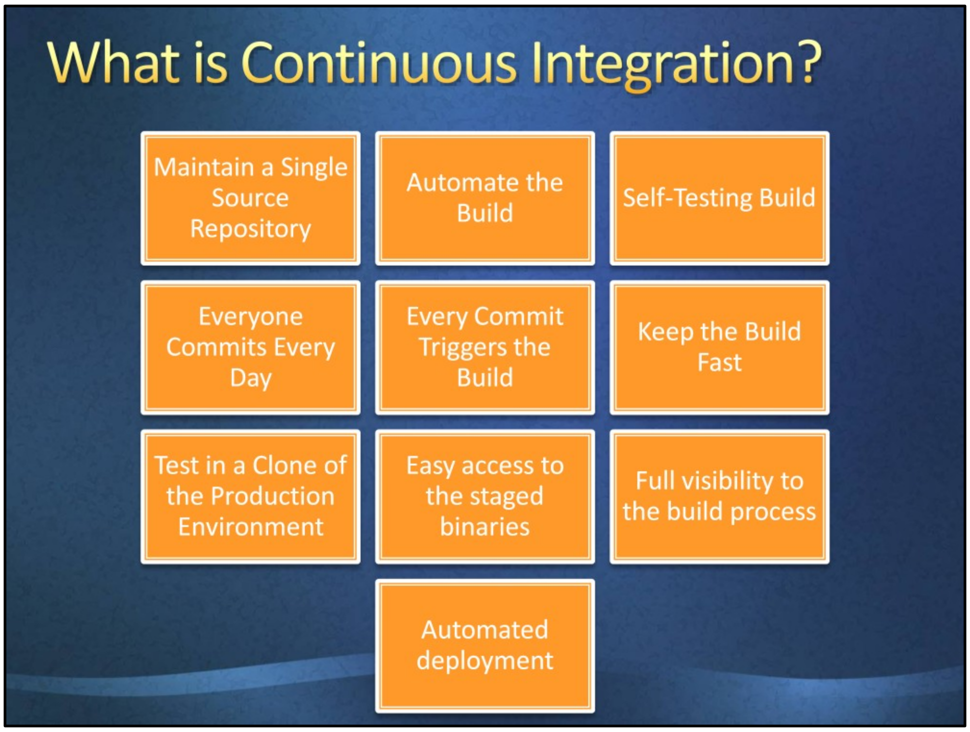
Today many organizations are trying to incorporate Continuous Integration into software development process because years of experience has taught everyone leaving integration to the end of a project is a risk.

-History of Continuous Integration – We will look at how Continuous Integration came to life.
-Benefits of Continuous Integration – We will look at how Continuous Integration can benefit the development, managers and customer relations.
-Theory of Continuous Integration – We will look at high level overview of what pieces are involved in creating Continuous Integration before we go into more detailed discussion.
-Tools of Continuous Integration – Tools are important part of Continuous Integration because tools help manage many of the overheads caused by achieving Continuous Integration.



History of Continuous Integration

-Many of the practices of Continuous Integration came from XP (Extreme) programming development process as necessity
-Continuous Integration takes into account many of the Best Practices practiced in XP programming
-Martin Fowler promoted Continuous Integration process and along came open source product called CruiseControl from Thoughtworks
-SOA triggers the need for integration platform.



What is Continuous Integration?

Maintain a Single Source Repository

Automate the Build

Self-Testing Build

Everyone Commits Every Day

Every Commit Triggers the Build

Keep the Build Fast

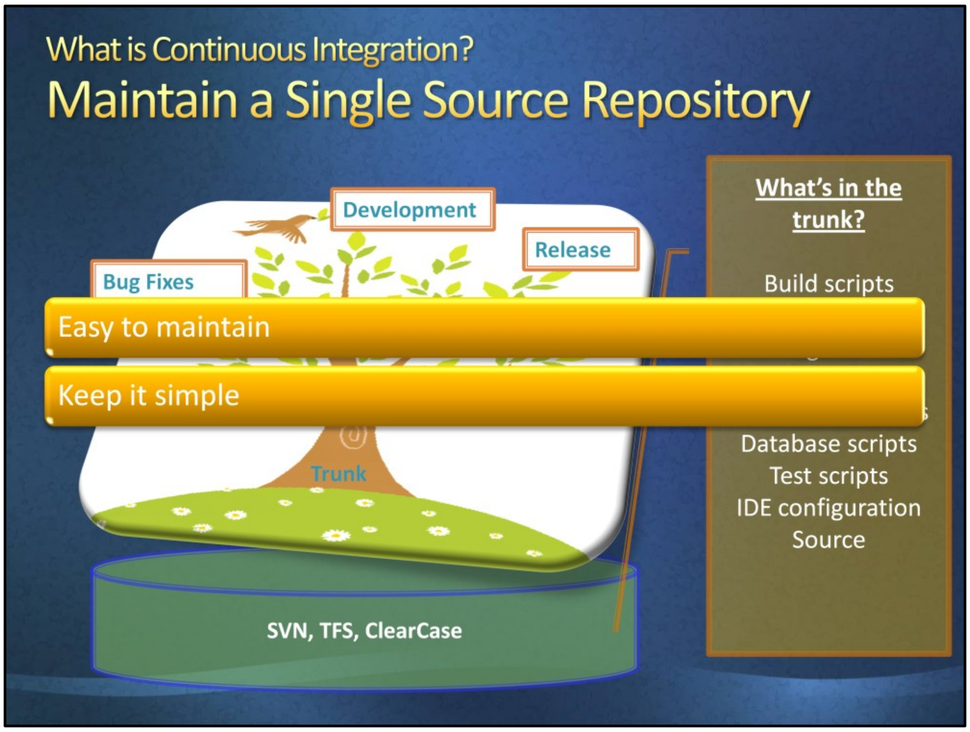
Test in a Clone of the Production Environment

Easy access to the staged binaries

Full visibility to the build process

Automated deployment

Here is the overview of the practices of Continuous Integration



What is Continuous Integration?

Maintain a Single Source Repository

Development
Release
Bug Fixes
Trunk
SVN, TFS, ClearCase

What's in the trunk?

Build scripts

Database scripts
Test scripts
IDE configuration
Source

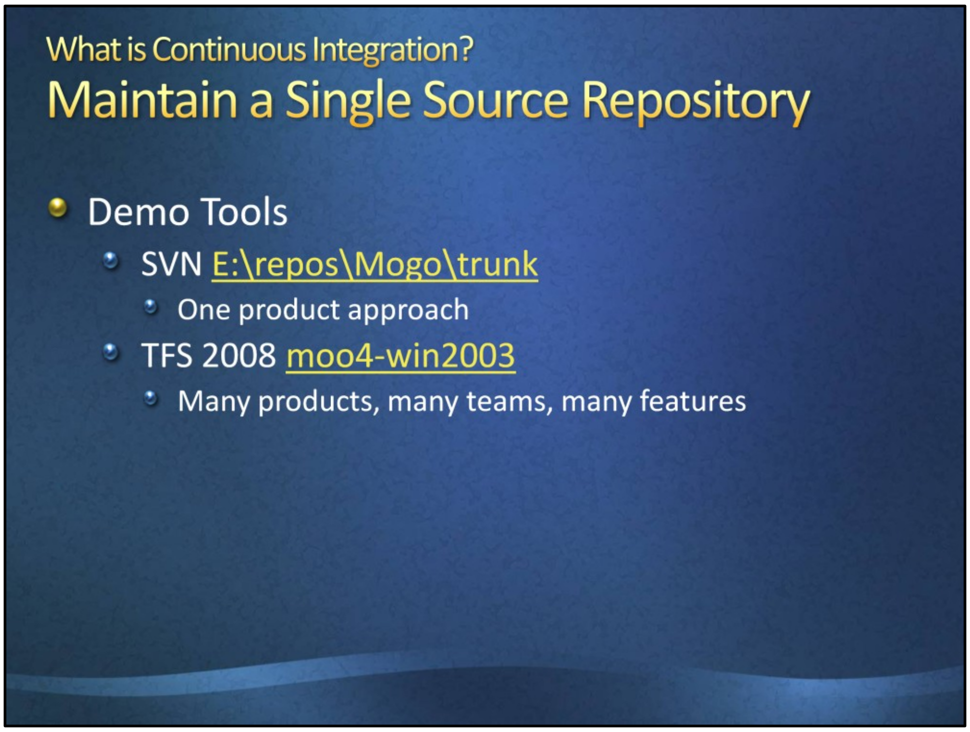
Easy to maintain

Keep it simple

-Need to pick one of Source Code Management tools like: Subversion, Team Foundation Server, ClearCase
-Include test scripts, configuration files, database schema and scripts, build scripts, install scripts, third party libraries, and IDE configuration.
-Keep the branches to minimum (bug fixes and maintenance, release, and experiments)

Pros:
1) Sources are centralized and thus easy to maintain
2) Everything is in repository and everyone knows where everything is

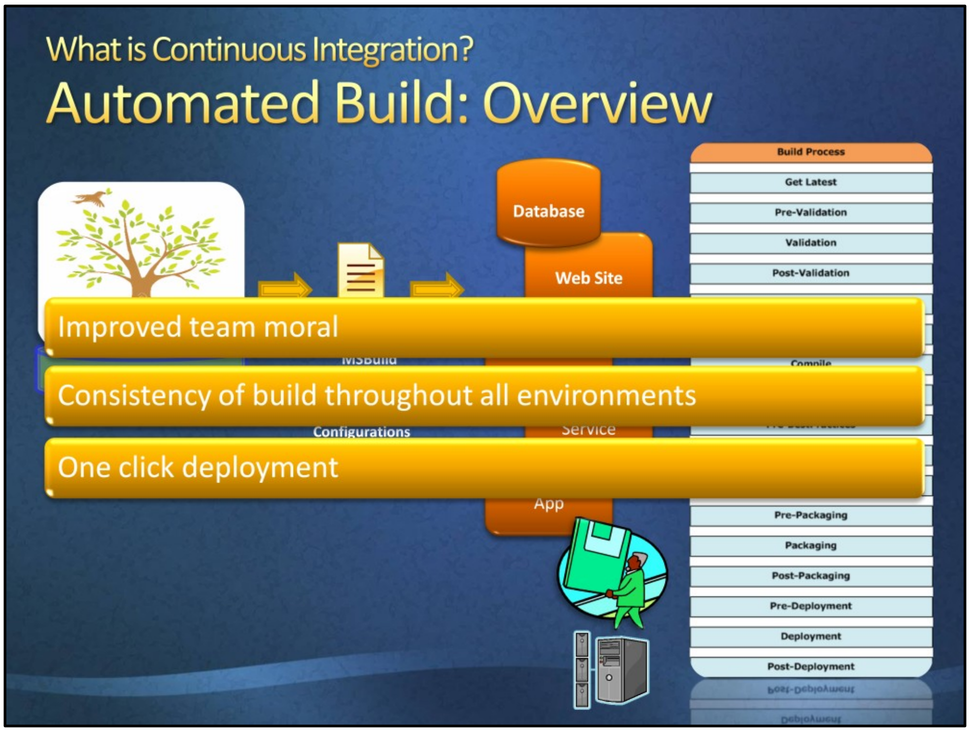
Cons:
1) Choosing right source control can be difficult because choosing source control depends on the needs of the organization



What is Continuous Integration?

Maintain a Single Source Repository

- Demo Tools
  - SVN E:\repos\Mogo\trunk
    - One product approach
  - TFS 2008 moo4-win2003
    - Many products, many teams, many features



What is Continuous Integration?

Automated Build: Overview

Improved team moral

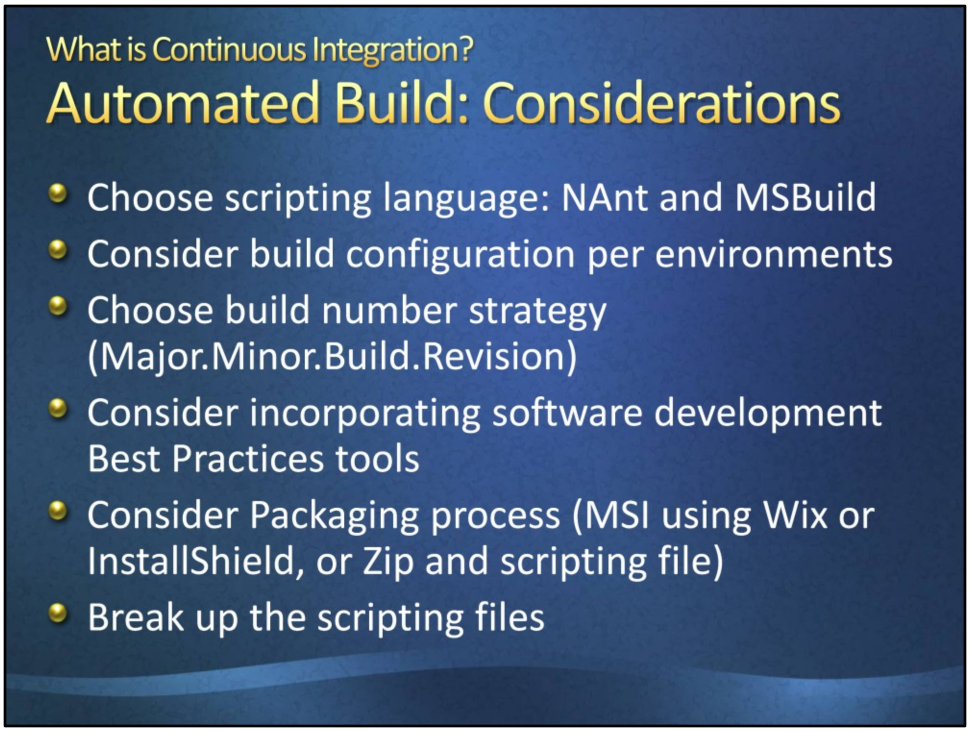
Consistency of build throughout all environments

One click deployment

-Sources into running system using build script such as Ant, NAnt, and MSBuild.
-Including EVRYTHING in the build things like database schema, database scripts, web site and web service deployment.
-Anyone should be able to get the latest source from the repository and execute build on the new machine and have system running.

Pros:
1) Single command converts sources into system.
2) New developers are ready to learn and develop right away because build is automated.
3) Every developers' environments are consistent.

Cons:
1) Introduces scripting language
2) Need to understand and learn new tool sets.



What is Continuous Integration?

Automated Build: Considerations

- Choose scripting language: NAnt and MSBuild
- Consider build configuration per environments
- Choose build number strategy (Major.Minor.Build.Revision)
- Consider incorporating software development Best Practices tools
- Consider Packaging process (MSI using Wix or InstallShield, or Zip and scripting file)
- Break up the scripting files

Choose a common scripting language for build:

NAnt:
1) Has very good open community support
2) Been around and used for long time

MSBuild:
1) MSBuild is constantly improved by Microsoft on every major framework release
2) Microsoft has plan to MSBuild-enable all development products (web and web service projects, WCF, WPF, Entity Framework, Silverlight, BizTalk)
3) Support parallel processing

Consider build configuration per environments:

1) How to handle local build vs. build server build?
2) How to build for Dev, QA, UAT and LT?
3) How to enable and disable specific build tasks?

Choose build number strategy because:

1) Branching based on build label
2) Associating work item to build number label
3) Packaging and deploy based on build number label

Consider incorporating software development Best Practices tools:

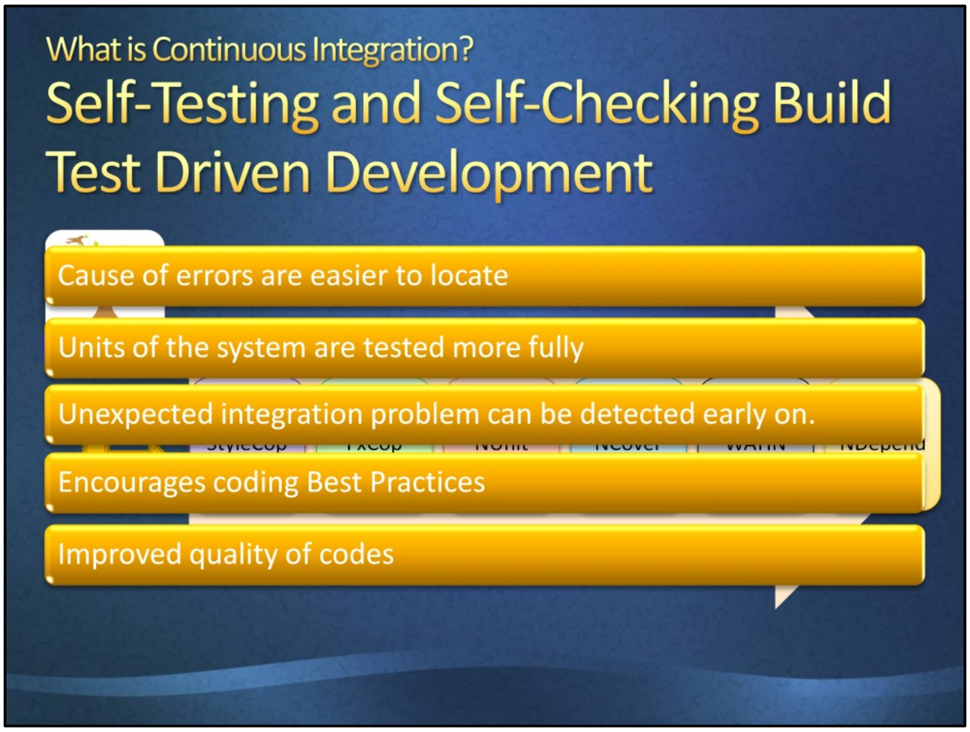
1) FxCop to check Microsoft best practices of coding guidelines
2) Style to check coding style to improve readability
3) Code coverage to enforce better unit testing of the code
4) Code Analysis (NDepend, VSTS) to improve code quality
5) Unit testing to stabilizing the code and minimize introducing bugs

Consider Packaging process:

1) Creating MSI using WIX or InstallShield
2) Create Zip file
3) Creating MSI or Zip of scripting files, batch files, and binaries for deployment

Break up the scripting files:

1) Breaking up the scripting files into reusable and manageable files
2) Document the scripting files for future reuse


What is Continuous Integration?

Self-Testing and Self-Checking Build Test Driven Development

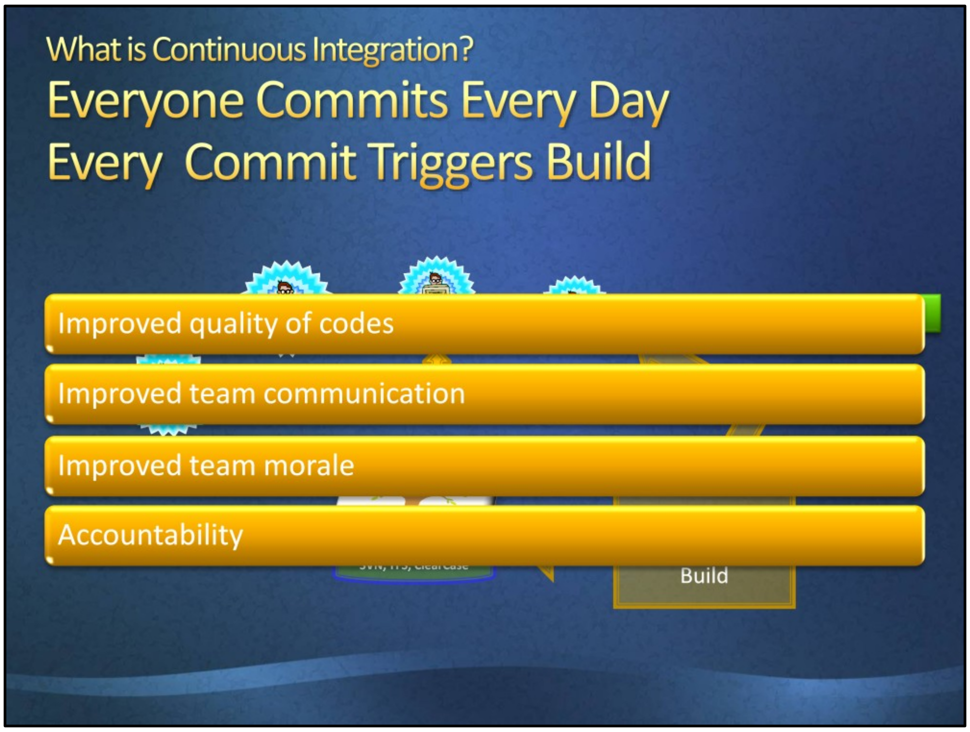
- Cause of errors are easier to locate
- Units of the system are tested more fully
- Unexpected integration problem can be detected early on.
- Encourages coding Best Practices
- Improved quality of codes

"The results of the case studies
indicate that the pre-release defect density of the four products decreased between 40% and
90% relative to similar projects that did not use the TDD practice. Subjectively, the teams
experienced a 15–35% increase in initial development time after adopting TDD…

From an efficacy perspective this increase in development time is offset by the by the reduced maintenance costs due to the improvement in quality (Erdogmus and Williams
2003), an observation that was backed up the product teams at Microsoft and IBM.
"
(http://channel9.msdn.com/posts/Peli/Experimental-study-about-Test-Driven-Development/)

-In Continuous Integration executing test is all part of build and build fails if the test fails.
- XUnit – JUnit, Nunit – for unit testing purpose
- Watir, Watin, Visual Studio Team Test – for quick functional testing (For web project do quick login to make sure it is running)

Pros:
1) Any changes made by the developers go through unit test and impact of code changes can be detected in early stage of development life cycle.
2) Promotes higher code quality
3) Developers are accountable to every check in that could potentially break build



What is Continuous Integration?

Everyone Commits Every Day Every Commit Triggers Build

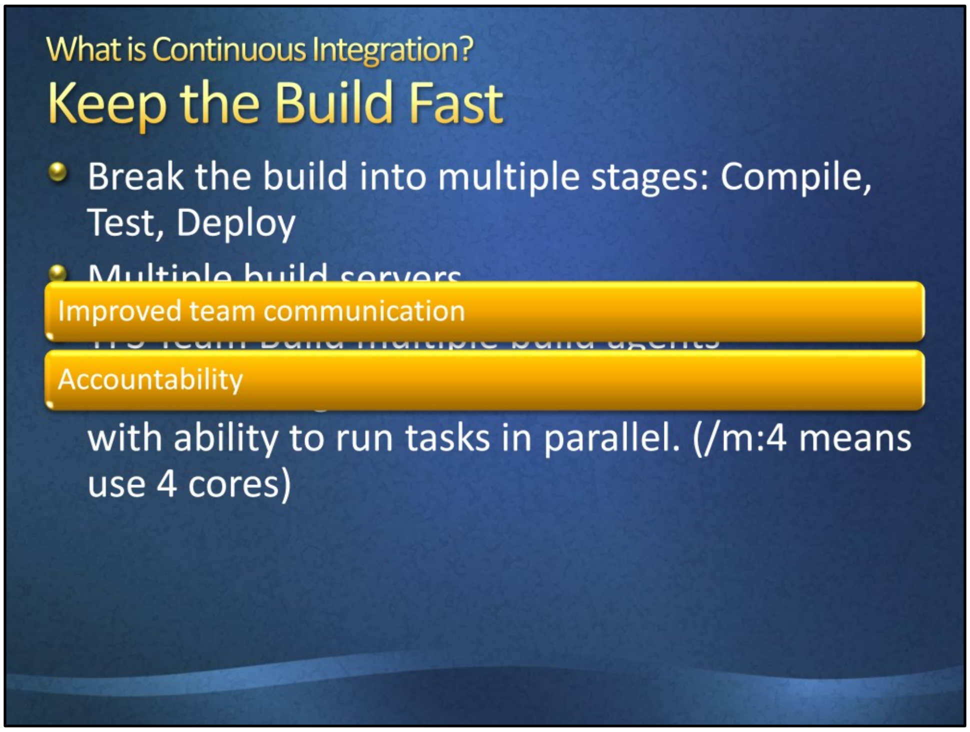
- Improved quality of codes
- Improved team communication
- Improved team morale
- Accountability

Pros:

1) Since build is self testing conflicts can be detected quickly.
2) Promotes communication between developers.
3) Allows the developers to break down works into smaller chunks.
4) Keeps the main build always stable.
5) Developers are accountable to failed build.

Cons:

1) Developers might hesitate to check in daily because meaningful work can not be checked in daily
2) Might require understanding of why commit and build.



What is Continuous Integration?

Keep the Build Fast

- Break the build into multiple stages: Compile, Test, Deploy
- Multiple build servers

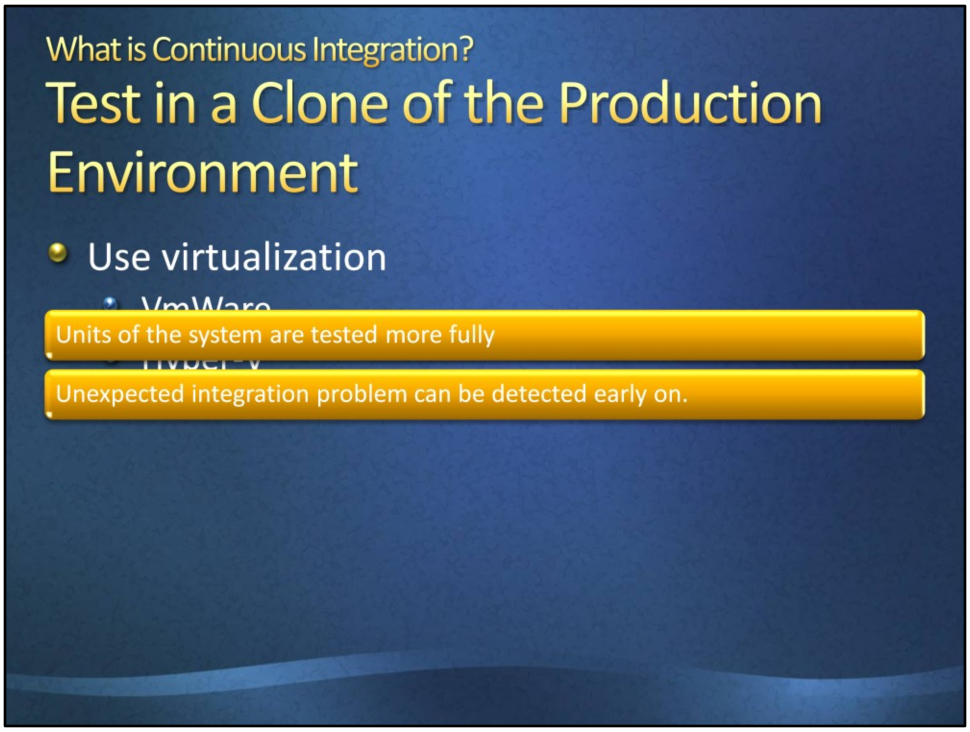
Improved team communication

Accountability

with ability to run tasks in parallel. (/m:4 means use 4 cores)

Pros:
1) Feedback is almost immediate.
2) Detect and fix errors quicker.
3) Gives confidence to other developers that if the codes are checked out the build would be good.

Cons:
1) Introduces some complexity



What is Continuous Integration?

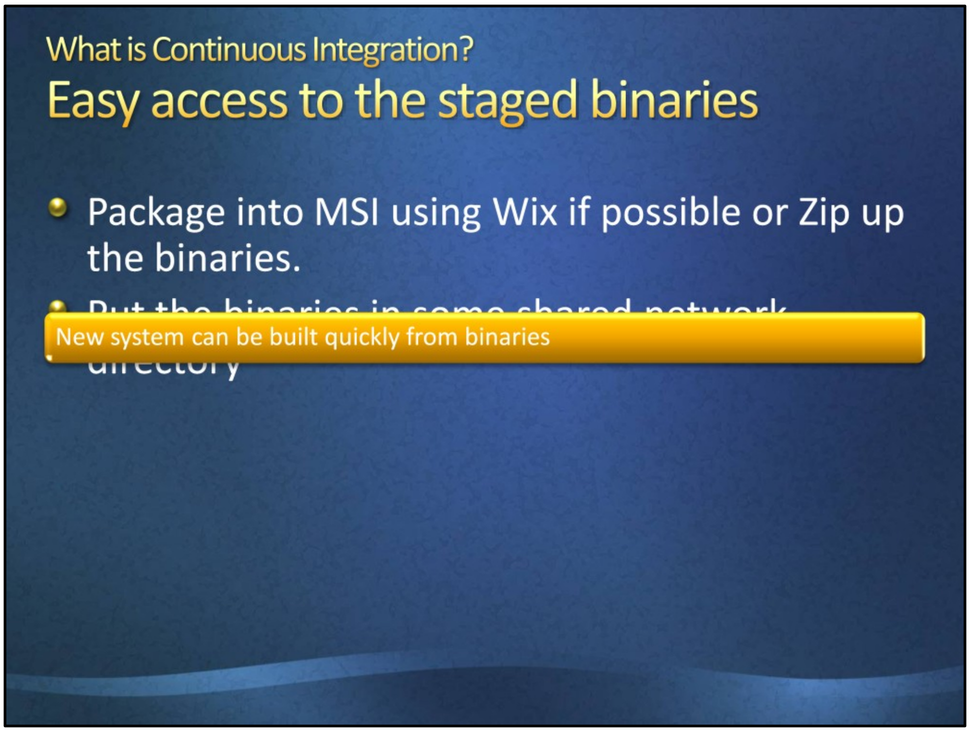
Test in a Clone of the Production Environment

- Use virtualization
  - VmWare
  - Hyper-V

Units of the system are tested more fully

Unexpected integration problem can be detected early on.

Cloning production environment can be very difficult task but virtualization of environments can help.



What is Continuous Integration?

Easy access to the staged binaries

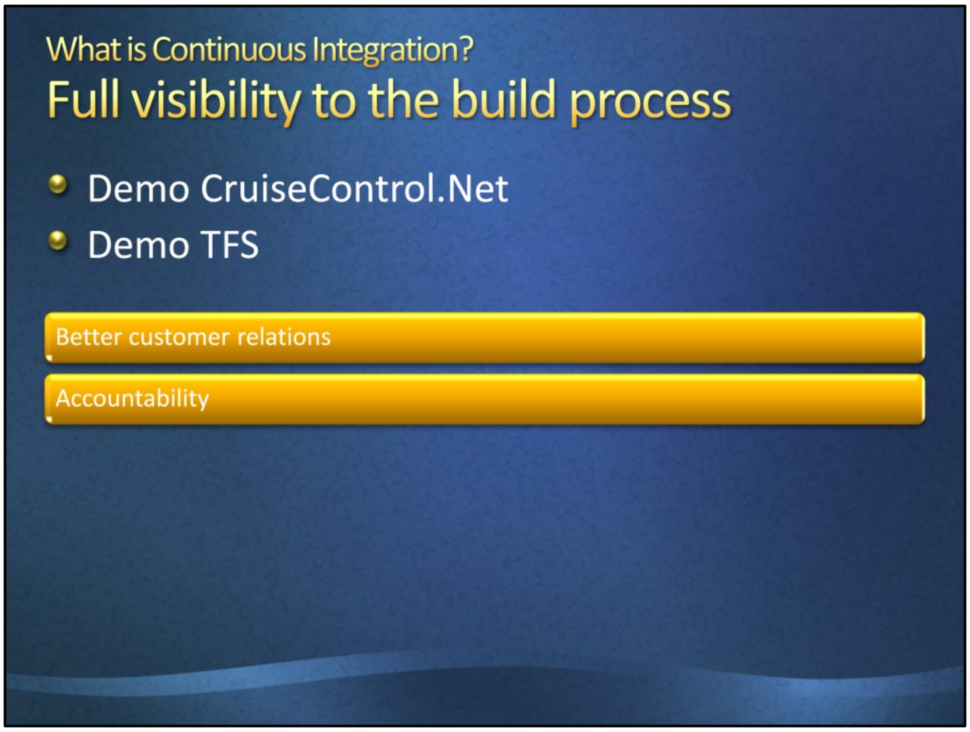
- Package into MSI using Wix if possible or Zip up the binaries.
- Put the binaries in some shared network directory

New system can be built quickly from binaries

-It is common practice in any software development to output binaries into common location where everyone can get to.

Pros:
1) MSI makes easy to deploy to any windows operating system
2) Easy to upgrade, repair and uninstall.

Cons:
1) Wix adds complexity to the packaging process



What is Continuous Integration?

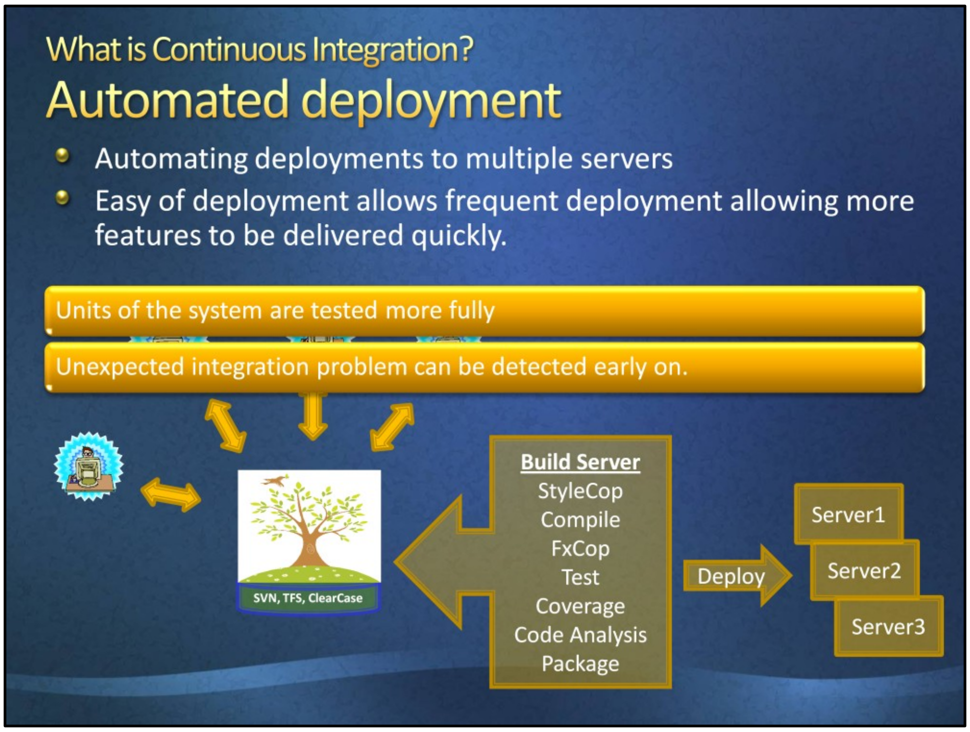
Full visibility to the build process

- Demo CruiseControl.Net
- Demo TFS

Better customer relations

Accountability

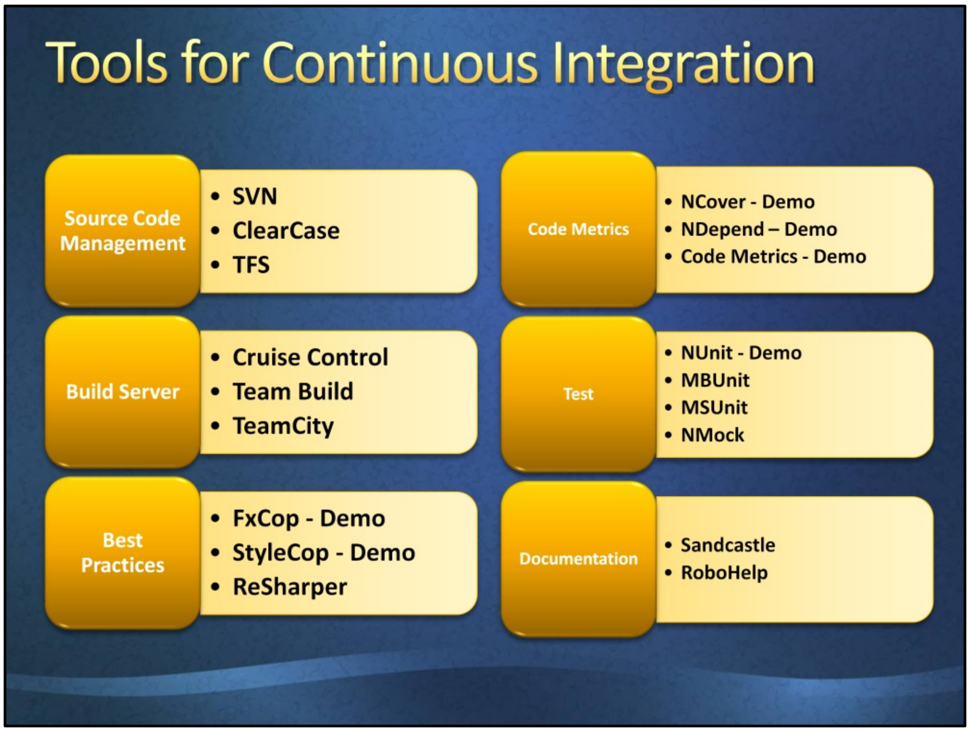
It is all about communication
Allows everyone to view what changes are made
Allows sneak preview into what systems are built


What is Continuous Integration?

Automated deployment

- Automating deployments to multiple servers
- Easy of deployment allows frequent deployment allowing more features to be delivered quickly.

Useful tools:
MSI deployment using psexec.exe
MSBuild http://msbuildtasks.tigris.org/
MSBuild http://www.codeplex.com/sdctasks
MSDeploy http://blogs.iis.net/msdeploy/archive/2008/01/22/welcome-to-the-web-deployment-team-blog.aspx



Tools for Continuous Integration

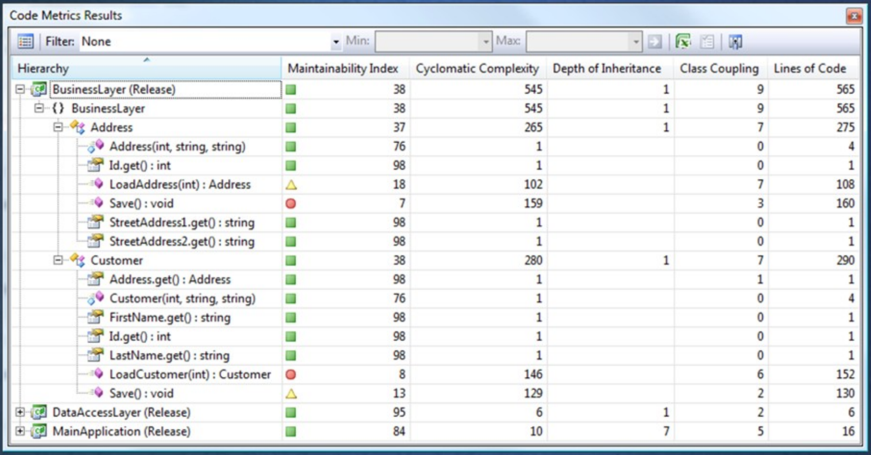
Source Code Management	• SVN • ClearCase • TFS
Build Server	• Cruise Control • Team Build • TeamCity
Best Practices	• FxCop - Demo • StyleCop - Demo • ReSharper
Code Metrics	• NCover - Demo • NDepend – Demo • Code Metrics - Demo
Test	• NUnit - Demo • MBUnit • MSUnit • NMock
Documentation	• Sandcastle • RoboHelp

Choice of tools to use in Continuous Integration has improved drastically in last few years; tools are becoming easy to use and powerful, and nicely integrated into GUI based development tools like Visual Studio or Eclipse and usability of tools are becoming seamless.

The key considerations for choosing right tools to use are as follow:

1) Command line based to be used in scripting language for automation purpose?
2) Built into Visual Studio?
3) Does it need to be preinstalled before the execution?
4) Can it be reused?
5) How easy is it to learn and use?
6) Product is maintained and supported?

Code Metrics



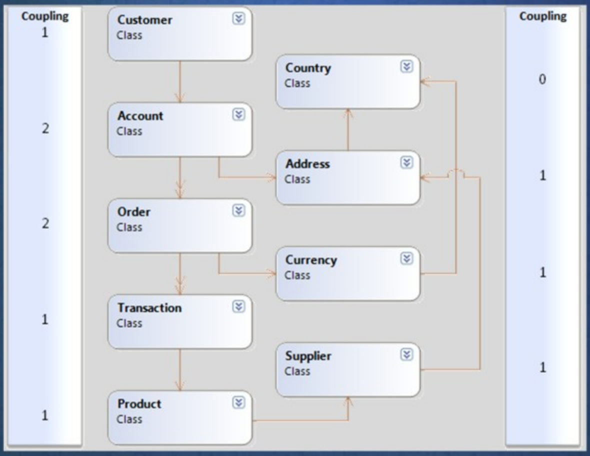
Code Metrics Results

Filter: None Min: Max:

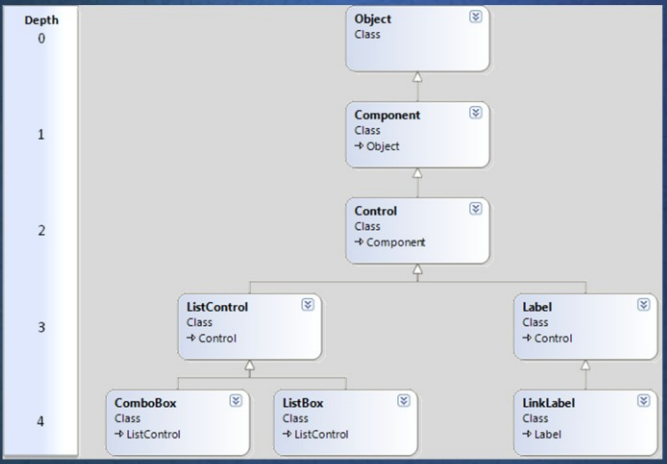
Hierarchy	Maintainability Index	Cyclomatic Complexity	Depth of Inheritance	Class Coupling	Lines of Code
BusinessLayer (Release)	🟩 38	545	1	9	565
{ } BusinessLayer	🟩 38	545	1	9	565
Address	🟩 37	265	1	7	275
Address(int, string, string)	🟩 76	1		0	4
Id.get() : int	🟩 98	1		0	1
LoadAddress(int) : Address	△ 18	102		7	108
Save() : void	● 7	159		3	160
StreetAddress1.get() : string	🟩 98	1		0	1
StreetAddress2.get() : string	🟩 98	1		0	1
Customer	🟩 38	280	1	7	290
Address.get() : Address	🟩 98	1		1	1
Customer(int, string, string)	🟩 76	1		0	4
FirstName.get() : string	🟩 98	1		0	1
Id.get() : int	🟩 98	1		0	1
LastName.get() : string	🟩 98	1		0	1
LoadCustomer(int) : Customer	● 8	146		6	152
Save() : void	△ 13	129		2	130
DataAccessLayer (Release)	🟩 95	6	1	2	6
MainApplication (Release)	🟩 84	10	7	5	16

🟩	High Maintainability	Between 20 and 100 inclusive
△	Moderate Maintainability	Between 10 and 19 inclusive
●	Low Maintainability	Between 0 and 9 inclusive

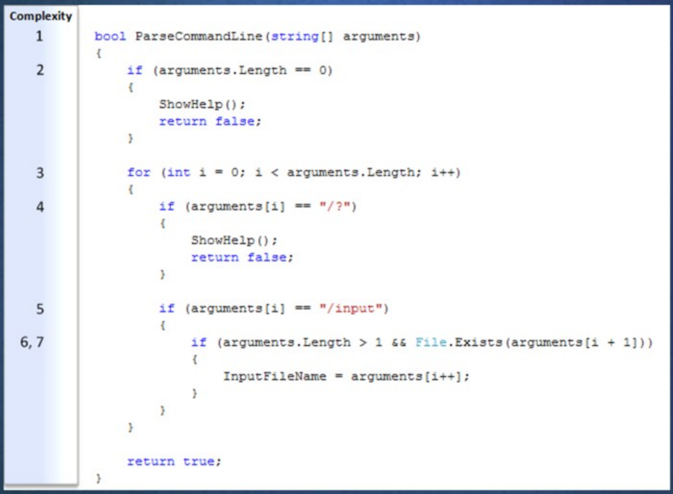
Class Coupling



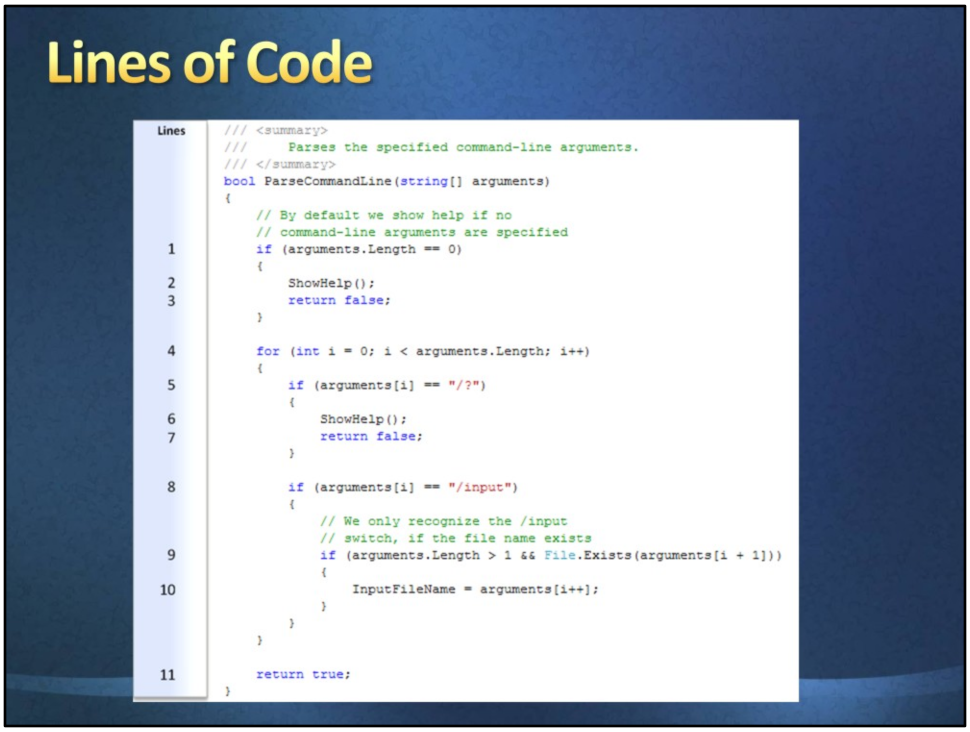
Depth of Inheritance



Cyclomatic Complexity



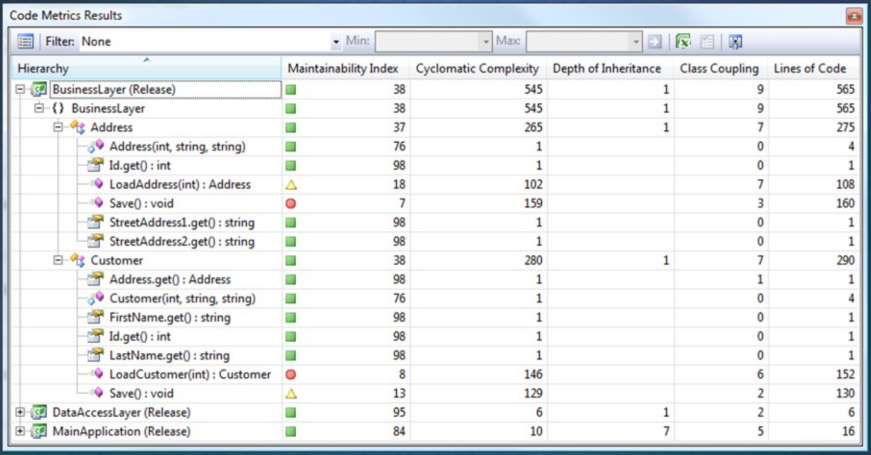
```
Complexity
   1        bool ParseCommandLine(string[] arguments)
            {
   2            if (arguments.Length == 0)
                {
                    ShowHelp();
                    return false;
                }

   3            for (int i = 0; i < arguments.Length; i++)
                {
   4                if (arguments[i] == "/?")
                    {
                        ShowHelp();
                        return false;
                    }

   5                if (arguments[i] == "/input")
                    {
  6, 7                  if (arguments.Length > 1 && File.Exists(arguments[i + 1]))
                        {
                            InputFileName = arguments[i++];
                        }
                    }
                }

                return true;
            }
```



Lines of Code

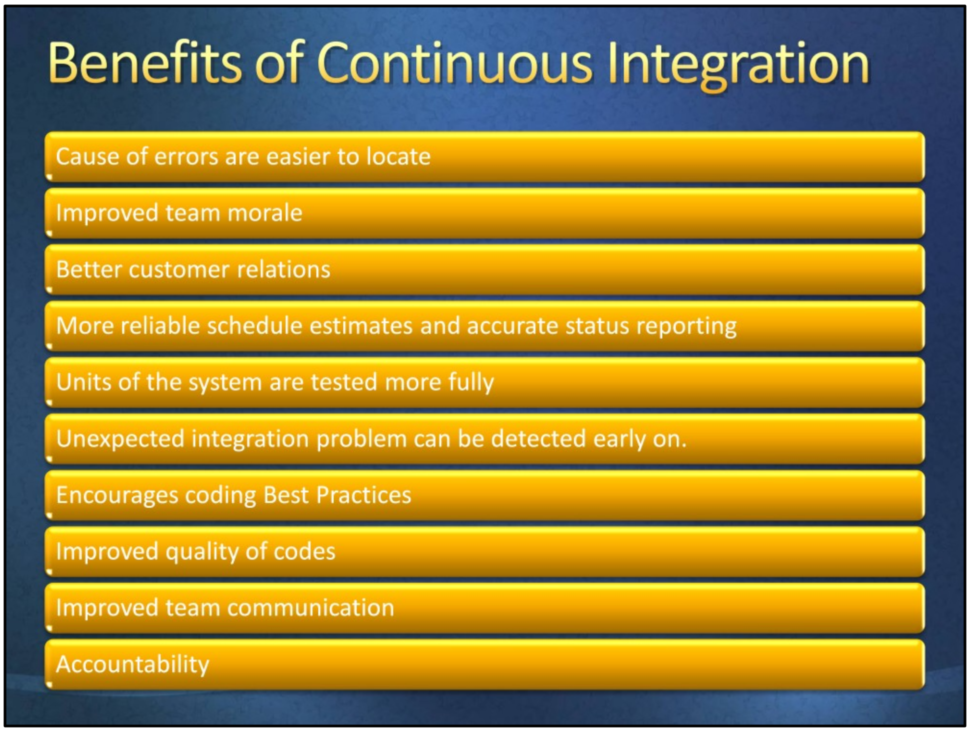
```
Lines
        /// <summary>
        ///     Parses the specified command-line arguments.
        /// </summary>
        bool ParseCommandLine(string[] arguments)
        {
            // By default we show help if no
            // command-line arguments are specified
  1         if (arguments.Length == 0)
            {
  2             ShowHelp();
  3             return false;
            }

  4         for (int i = 0; i < arguments.Length; i++)
            {
  5             if (arguments[i] == "/?")
                {
  6                 ShowHelp();
  7                 return false;
                }

  8             if (arguments[i] == "/input")
                {
                    // We only recognize the /input
                    // switch, if the file name exists
  9                 if (arguments.Length > 1 && File.Exists(arguments[i + 1]))
                    {
 10                     InputFileName = arguments[i++];
                    }
                }
            }

 11         return true;
        }
```

Code Metrics



Hierarchy	Maintainability Index	Cyclomatic Complexity	Depth of Inheritance	Class Coupling	Lines of Code
BusinessLayer (Release)	🟩 38	545	1	9	565
{ } BusinessLayer	🟩 38	545	1	9	565
Address	🟩 37	265	1	7	275
Address(int, string, string)	🟩 76	1		0	4
Id.get() : int	🟩 98	1		0	1
LoadAddress(int) : Address	⚠ 18	102		7	108
Save() : void	🔴 7	159		3	160
StreetAddress1.get() : string	🟩 98	1		0	1
StreetAddress2.get() : string	🟩 98	1		0	1
Customer	🟩 38	280	1	7	290
Address.get() : Address	🟩 98	1		1	1
Customer(int, string, string)	🟩 76	1		0	4
FirstName.get() : string	🟩 98	1		0	1
Id.get() : int	🟩 98	1		0	1
LastName.get() : string	🟩 98	1		0	1
LoadCustomer(int) : Customer	🔴 8	146		6	152
Save() : void	⚠ 13	129		2	130
DataAccessLayer (Release)	🟩 95	6	1	2	6
MainApplication (Release)	🟩 84	10	7	5	16

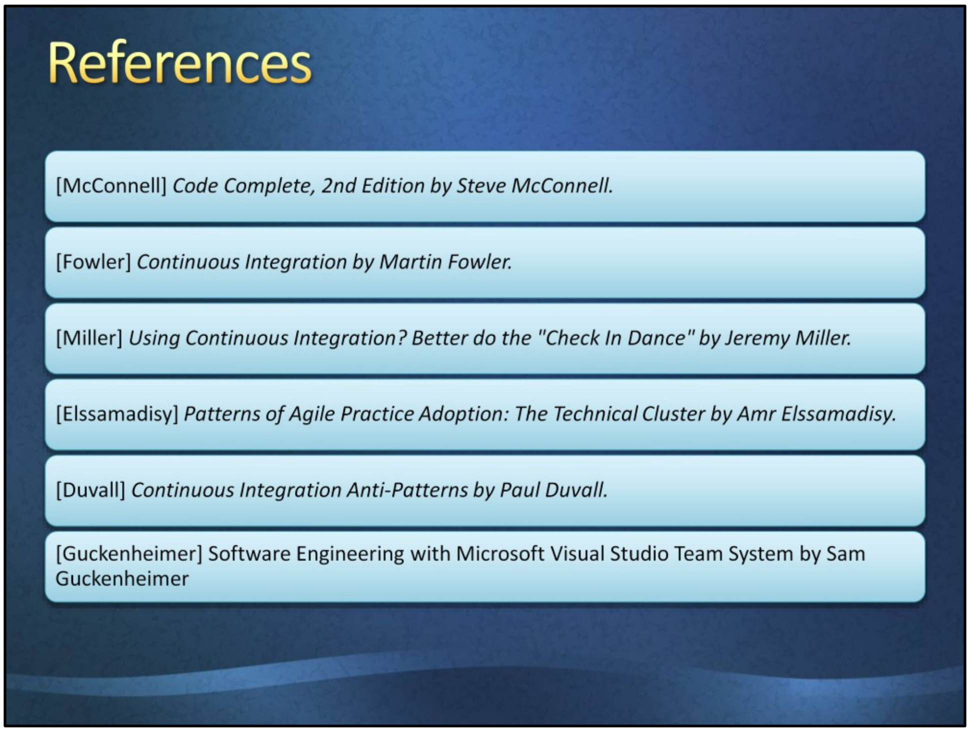
🟩	High Maintainability	Between 20 and 100 inclusive
⚠	Moderate Maintainability	Between 10 and 19 inclusive
🔴	Low Maintainability	Between 0 and 9 inclusive



Benefits of Continuous Integration

- Cause of errors are easier to locate
- Improved team morale
- Better customer relations
- More reliable schedule estimates and accurate status reporting
- Units of the system are tested more fully
- Unexpected integration problem can be detected early on.
- Encourages coding Best Practices
- Improved quality of codes
- Improved team communication
- Accountability

Recapping the benefits mentioned on each of the Continuous Integration Practice



References

[McConnell] *Code Complete, 2nd Edition by Steve McConnell.*

[Fowler] *Continuous Integration by Martin Fowler.*

[Miller] *Using Continuous Integration? Better do the "Check In Dance" by Jeremy Miller.*

[Elssamadisy] *Patterns of Agile Practice Adoption: The Technical Cluster by Amr Elssamadisy.*

[Duvall] *Continuous Integration Anti-Patterns by Paul Duvall.*

[Guckenheimer] Software Engineering with Microsoft Visual Studio Team System by Sam Guckenheimer

Questions and Answers