FLAMES®
Simulation Framework

OVERVIEW OF THE FLAMES® SIMULATION FRAMEWORK

FLAMES® IS A FAMILY OF COMMERCIAL OFF-THE-SHELF (COTS) SOFTWARE PRODUCTS THAT PROVIDE A FRAMEWORK FOR DEVELOPING CUSTOM CONSTRUCTIVE AND VIRTUAL SIMULATIONS AND INTERFACES BETWEEN LIVE, VIRTUAL, AND CONSTRUCTIVE (LVC) SIMULATIONS. FLAMES DRAMATICALLY REDUCES THE TIME AND MONEY REQUIRED TO DEVELOP AND MAINTAIN SIMULATIONS WITH UNMATCHED CAPABILITIES.



PRECISE SYSTEMS
SCALE YOUR FUTURE



OVERVIEW OF THE FLAMES SIMULATION FRAMEWORK

This paper begins with a high-level description of FLAMES and an overview of some of the reasons why you should use FLAMES to develop your simulations. More detailed information about FLAMES is then provided followed by an overview of the process of developing a FLAMES-based simulation.

WHAT IS FLAMES?

First of all, FLAMES is NOT a simulation. It does not simulate anything. Rather, FLAMES is a simulation "framework" – a software system that serves as the foundation for simulations.

FLAMES provides the infrastructure that is needed by every simulation in a framework that is independent of any specific simulation. This infrastructure includes applications for scenario editng and execution, a scenario database manager, a comprehensive application programming interface, development tools, and much more.

Components, the software that simulates systems that exist in the real world, reside completely external to FLAMES in "plugins".

FLAMES provides a true "plug-and-play" architecture. Component plugins are integrated into FLAMES applications automatically when they start.

Different simulations are created by creating different component plugins and using them in various combinations. All simulations are built on the existing FLAMES framework.

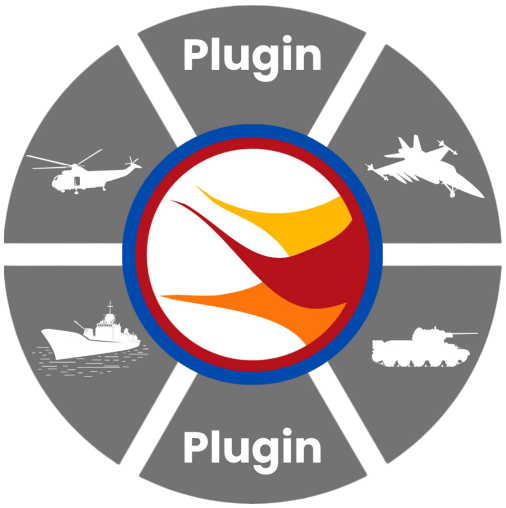


Figure 1: FLAMES® Component Architecture

WHY USE FLAMES?

This section describes some of the ways that using FLAMES to develop your simulation can save oney, save time, reduce risk, and help you get a more capable simulation.

FLAMES SIMPLIFIES SIMULATION DESIGN

Erich Gamma, one of the "fathers" of object-oriented programming, wrote in his book, "Design Paterns: Elements of Reusable Object-Oriented Software" (Addison Wesley, 1995):

"Designing object-oriented software is hard, and designing reusable object-oriented software is even harder. You must find pertinent objects, factor them into classes at the right granularity, define class interfaces, and inheritance hierarchies, and establish key relationships among them."

FLAMES dramatically shortens and simplifies the design phase of your simulation development because most of the design is complete and fully implemented. The "hard" work described by Mr. Gamma is done. The design of every aspect of FLAMES and the classes defined by FLAMES has been proven in over 30 years of use in over 100 different simulations.

FLAMES REDUCES THE AMOUNT OF SOFTWARE YOU NEED TO DEVELOP

The majority of the software in any FLAMES-based simulation is provided by the FLAMES framework. This is software that you do not need to develop, test, or maintain. It works reliably and efficiently the first day you start your simulation development project.

In addition, FLAMES includes several GitHub repositories that contain the complete source code to component classes that simulate dozens of different types of real world systems. You can use these classes as they are, or you can modify them or use them as the starting point for developing new classes. This further reduces the amount of software that you need to develop.

FLAMES SIMPLIFIES THE SOFTWARE YOU DO NEED TO DEVELOP

All FLAMES component classes inherit one of the base classes defined by FLAMES. These welldesigned and well-documented base classes serve as a guide as you develop your classes and greatly simplify your software development. In addition, FLAMES base classes include many built-in capabilities that are automatically available in all component classes which further implifies your software development.

The plug-and-play architecture of FLAMES eliminates the effort required to integrate new classes and is yet another feature of FLAMES that simplifies software development.

FLAMES ALLOWS TRUE SOFTWARE REUSE

Properly developed FLAMES component plugins can be loaded in a FLAMES-based simulation in any combination. This means true software reuse is made possible. Components developed for one FLAMES-based simulation can easily be reused in another FLAMES-based simulation.

FLAMES REDUCES RISK

Large simulations are complex software systems. As is evidenced by many past and current simulation development programs, developing a simulation from “scratch” is a high-risk endeavor that is likely to fail to produce the intended results – or to fail completely. Experienced software developers know this well. One such developer is Grady Booch, another one of the fathers of object-oriented programming. In his book, “Object Oriented Design: With Applications” (The Benjamin/Cummings Publishing Company, Inc., 1991), Booch wrote:

“A complex system that works is invariably found to have evolved from a simple system that worked… A complex system designed from scratch never works and cannot be patched up to make it work. You have to start over, beginning with a working simple system.”

FLAMES not only reduces the amount of time and money required to develop a complex simulation, it also dramatically reduces the risk in the development effort. FLAMES already exists, it defines the design of the simulation, and, most importantly, it works.

FLAMES-BASED SIMULATIONS HAVE MORE CAPABILITIES

FLAMES has been used by Ternion and other organizations for over 30 years to develop over 100 different simulations. This varied and extensive use of FLAMES has provided Ternion with an abundance of requirements, use cases, and feedback in the development of FLAMES. In addition, Ternion is a commercial software company, and the development of FLAMES was not constrained by the requirements of any one customer or program. As a result, FLAMES provides more capabilities than any other simulation framework. These capabilities are available on day one to any simulation developed using FLAMES.

WHAT IS INCLUDED WITH FLAMES?

This section describes the primary items included with FLAMES, which are the FLAMES Launcher, the FLAMES Developer, the FLAMES Engine, FLAMES Options, FLAMES Content, FLAMES Content Source, and Documentation and Training Videos.

THE FLAMES LAUNCHER

The FLAMES Launcher is a free desktop application that provides a single, easy-to-use interface for purchasing, downloading, installing, and executing FLAMES products and FLAMES content. The Launcher includes the following sections:

The Store is the place to find and purchase FLAMES products and FLAMES content. Some products are FREE, and all content is FREE. The Store is available in the FLAMES Launcher and on the FLAMES website (store.flamesframework.com).

The Library section of the Launcher allows you to view, download, and install the FLAMES products and content that you have purchased from the FLAMES Store.

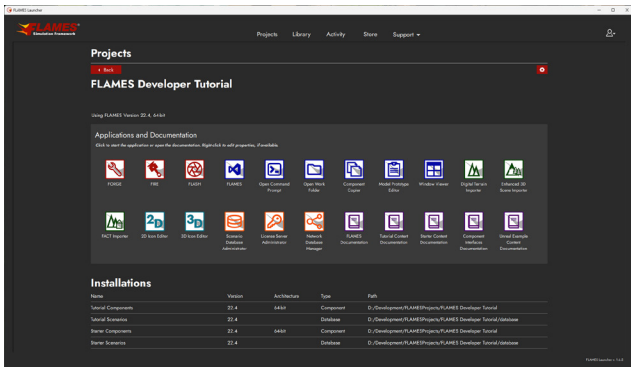


Figure 2: FLAMES Launcher Project

The Projects section of the Launcher manages your FLAMES “projects”. A project organizes a desired set of FLAMES content in a single place and provides a simple and convenient interface or starting FLAMES applications, developing FLAMES component plugins, and viewing FLAMES documentation.

THE FLAMES DEVELOPER

The FLAMES Developer is the software development kit (SDK) and the tools you need to develop component plugins containing classes that simulate the behavior of any real-world system in accordance with your exact requirements. The classes in a plugin can also extend the functionality of the FLAMES Engine. More information on the FLAMES Developer is provided later in this paper under “Developing a FLAMES-Based Simulation”.

Licenses to the FLAMES Developer are FREE and can be installed on as many computers as you desire, so you can equip your entire software development team with the tools they need to create the simulations that you want.

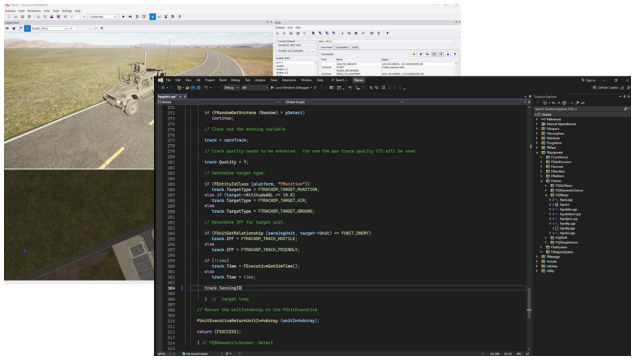


Figure 3: FLAMES Developer

The FLAMES component plugins that you develop can be distributed for free and used on any computer on which the FLAMES Engine is properly licensed.

THE FLAMES ENGINE

The FLAMES Engine is a set of applications (executable programs) that allow you to create, execute, visualize, and control FLAMES scenarios. When they start, these applications dynamically load and integrate any specified set of FLAMES component plugins to allow you to simulate almost any system in almost any scenario imaginable. The primary applications of the FLAMES Engine are FORGE™, FIRE™, and FLASH™.

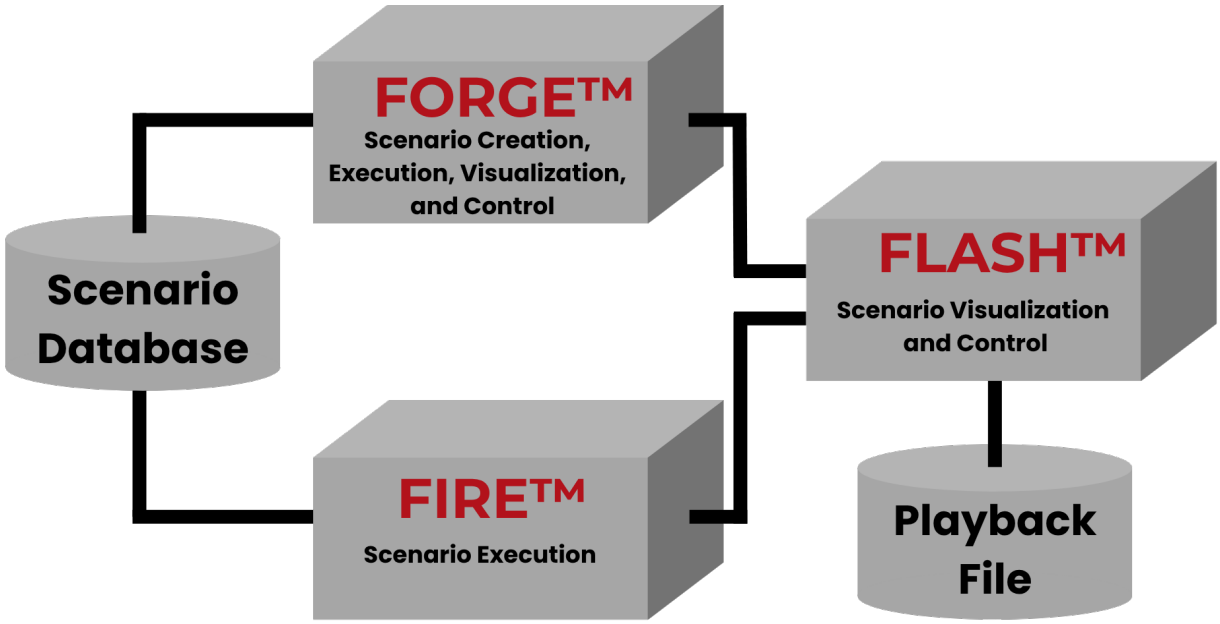


Figure 4: FLAMES Engine Applications

FORGE

The primary application used to create, execute, and visualize FLAMES scenarios. FORGE also allows you to control scenarios as they execute. FORGE employs a friendly graphical user interface to greatly simplify the task of entering scenario data. It stores the data you enter in the FLAMES multi-user scenario database. 2D and 3D displays allow you to visualize your scenario as you create and execute it.

FIRE

Another application that can be used to execute FLAMES scenarios. FIRE retrieves a scenario from the FLAMES database and executes it. FIRE executes in batch mode and contains no graphics or built-in user interface. This allows scenarios to be executed more quickly and to be executed on servers or virtual machines. Some FLAMES options can be used only in FIRE.

FLASH

Used to visualize the activity in a scenario after it has been executed using the same 2D and 3D graphic displays available in FORGE. On or more instances of FLASH can also be used to visualize and control players in a scenario from a remote location as the scenario executes in FORGE or FIRE.

A trial version of the FLAMES Engine is included for FREE with the FLAMES Developer. The trial version is fully functional, but it has limited capacity, and it may be used only for short-term evaluation purposes or to debug and test the software developed using the FLAMES Developer. When you are ready to use the FLAMES Engine in some other capacity, you will need to purchase an annual subscription to a full, unrestricted license.

FLAMES OPTIONS

FLAMES includes several optional products that extend the functionality of the FLAMES Engine to address specialized simulation requirements.

UNREAL ENGINE OPTION.

Tightly integrates a game developed using Epic Games' Unreal Enginedirectly into a FLAMES simulation. This option allows all the power of both FLAMES and UnrealEngine to be employed to create constructive and virtual simulations with unmatchedcapabilities. Fully functional, full capacity licenses to the Unreal Engine option are free.

ENHANCED ANALYSIS OPTION.

Allows FIRE to use "experiment files" to automatically execute a scenario multiple times to perform complex parametric trade studies and Monte-Carlo analysis..

DISTRIBUTED INTERACTIVE SIMULATION (DIS) OPTION.

Allows FORGE and FIRE to exchange data with other simulations during scenario execution using the IEEE DIS protocol.

HIGH LEVEL ARCHITECTURE (HLA) OPTION.

Allows FORGE and FIRE to participate in HLA federations using a third-party HLA runtime infrastructure (RTI).

CHECKPOINT/RESTART OPTION.

Allows FIRE to collect all of the data in an executing scenario and periodically save a snapshot of the scenario in files called "checkpoint" files. Scenario execution can be restarted using the data saved in one of these checkpoint files.

CIGI OPTION.

Allows FORGE and FIRE to communicate with image generators using the Common Image Generator Interface (CIGI) protocol.

NETWORK DATABASE OPTION.

Allows FLAMES applications to access a FLAMES scenario database hosted on a remote server. Fully collaborative, multi-user scenario editing is also supported.

CIGI IG PLUGIN FOR UNREAL.

Allows the creation of CIGI- compatible IGs and integrated computer generated forces (CGF) simulations using FLAMES and Unreal Engine.

Trial versions of most FLAMES options are included for FREE with the trial version of the FLAMES Engine. You will need to purchase a license to an option to use it with the unrestricted version of the FLAMES Engine (except as noted above).

FLAMES CONTENT

FLAMES programs do not contain any software that simulates the behavior of real-world systems. As explained above, all such software resides in FLAMES component plugins.

One place to get FLAMES plugins is the content items available in the FLAMES Store. FLAMES content items usually include component plugins, example scenarios that use the components in the plugins, and documentation for the components and the scenarios. Some content items also include an Unreal Engine game in binary form that can be executed in FLAMES using the FLAMES Unreal Engine option.

FLAMES content allows you to become productive using FLAMES quickly and easily without having to do any software development. Currently, all content is available for FREE.

FLAMES CONTENT SOURCE

The FLAMESFramework organization on GitHub includes several repositories which contain source code for the component plugins available in the FLAMES Store. You can modify the source code to meet your requirements or use it as the starting point for totally new classes. Other repositories contain source code for useful tutorials and example programs.

FLAMES component classes can be writen in C or C++. Nearly all of the classes in the FLAMES GitHub repositories are in C++.

Complete Unreal Engine projects in source form are also available in the FLAMES Store for each of the Unreal games included in FLAMES content items. You can download these projects and use them as the starting point for developing your own FLAMES compatible Unreal games.

DOCUMENTATION AND TRAINING VIDEOS

FLAMES products and content are supplied with an extensive set of documentation that thoroughly explains how to use FLAMES and develop FLAMES-based simulations. In addition, dozens of training videos are available for free at flamesframework.com/training.

DEVELOPING A FLAMES-BASED SIMULATION

As has been explained, all FLAMES-based simulations are built on the existing FLAMES framework. In most cases, the process of developing a FLAMES-based simulation is reduced to developing the component classes that represent the real-world systems you want to simulate and assembling the classes into one or more FLAMES plugins. A high-level description of the process is provided below.

1. Choose an existing FLAMES component class that you wish to modify or use as the starting point for the development of a new class. Clone the FLAMESFramework GitHub repository that contains the source code of this class.

2. If you are creating a new class, start the FLAMES Component Copier using the FLAMES Launcher. Use this convenient tool to copy the existing class you chose in step 1 and give the class a new name. If you are going to modify the existing class directly, skip this step.

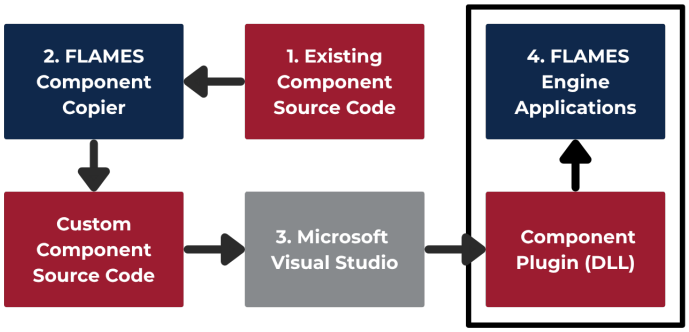


Figure 5: FLAMES Workflow

3. Start Microsoft Visual Studio using the FLAMES Launcher. Use Visual Studio to make the desired changes to the existing/new class and to create a FLAMES component plugin (i.e., a dynamic linked library (DLL)) that contains the class.

4. Start FORGE using the FLAMES Launcher. FORGE will automatically load and integrate the classes in the plugin. Your new class will be available to use in a scenario.

CONCLUSION

FLAMES dramatically reduces the time and money required to develop and maintain constructive and virtual simulations with unmatched capabilities. Download the FLAMES Developer for free and build your next simulation using FLAMES. Chances are, you'll never build another simulation any other way.

ABOUT FLAMES®

FLAMES is a family of commercial off-the-shelf (COTS) software products that provide a framework for developing custom constructive and virtual simulations and interfaces between live, virtual, and constructive (LVC) simulations. The optional integration with Unreal® Engine extends FLAMES to provide the ultimate framework for the creation of serious games and visually stunning, entity-level constructive and virtual simulations. For more information on FLAMES or to download the FREE FLAMES Developer, visit flamesframework.com.

ABOUT TERNION, A PRECISE SYSTEMS COMPANY

Ternion, a Precise Systems Company is the developer of FLAMES and an expert in developing custom, FLAMESbased simulations for government and commercial organizations worldwide. To learn more about Ternion's past projects and how Ternion can help you build your constructive and virtual simulations or build them for you, visit ternion.com.

FLAMES® Simulation Framework

TERNION® A PRECISE SYSTEMS COMPANY Simulations That Work℠


PRECISE
SYSTEMS
SCALE YOUR FUTURE