

What is EL4NET?

EL4NET is an extension library for the .NET framework 2.0 that helps implement and operate agile enterprise-class applications. It provides high application agility and rapid development by focusing on business object development, while deferring architectural and deployment choices to the application configuration. Development of enterprise-class distributed applications is facilitated by EL4NET's homogenous high-level programming model, which includes an extensible component framework.

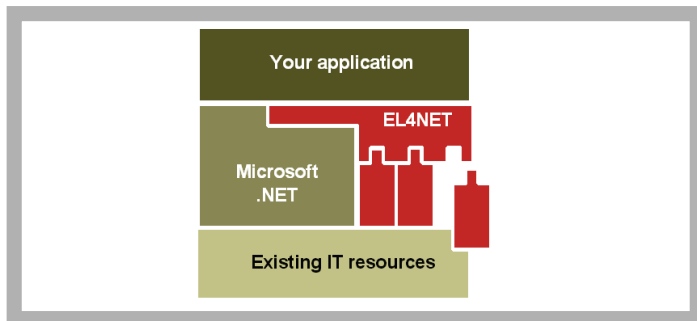


Figure 1: Basic idea: simplify and extend .NET

What are the benefits of EL4NET?

Developing with EL4NET has the following benefits (compared to pure .NET development):

- **Speed up application development:** EL4NET provides a state-of-the-art architecture model and follows best-practices, allowing you to focus your design efforts on the essential business concerns.
- **Embrace change/save costs:** All projects inevitably face specification or architectural changes. Depending on when this change occurs, the costs and risks can be very high. EL4NET minimizes such costs and risks by providing enough agility in the project architecture.
- **Separate concerns:** EL4NET allows you to effectively separate business from technical concerns by providing an interception layer and other AOP mechanisms.
- **Organize your configuration:** EL4NET provides an operation friendly configuration system that separates component wiring from application parameterization.
- **Choose between more remoting options:** EL4NET allows publishing components with various technologies like .NET remoting, ASP.NET Web Services, or Indigo.
- **Deploy flexibly:** EL4NET supports all popular application models: Fat clients, web applications, 3-tier rich client applications, etc. To ease development and improve testability, EL4NET allows deploying an application in different ways.

What does EL4NET consist of?

The EL4NET architecture toolbox contains:

- Lightweight inversion of control container
- Plugin infrastructure
- Configuration system and configuration tools
- Extended remoting support
- Flexible deployment support
- Logging abstraction (log4net and .NET diagnostics)

In the following sections, we describe the features of this toolbox.

Open component model

EL4NET defines a simple yet powerful extensible component model. This model leverages architectural patterns and best practices such as separation of concerns, separation of interface and implementation, inversion of control, component locator and layered architecture.

A component is the smallest application unit managed by EL4NET. Each component implements a contract interface, and has its own configuration. Components are discovered and constructed at runtime. The lifecycle and the activation model for a component are defined by its component type. EL4NET supports singleton, stateless and stateful components out-of-the-box.

Different components can either be co-located in the same appdomain or can be deployed in different appdomains/processes. This is transparent in the code, which treats invocation to co-located and remote components in the same way (uniform and location-independent access).

The component model can be extended in multiple ways. It allows defining and adding new component types without changing the framework. The new component types can take advantage of all the framework's features. An interception layer allows defining special purpose components (named invokers) for pre- and post-processing invocations to solve orthogonal technical concerns. Those invokers can be declared either in configuration or using .NET custom attributes.

Advantages: The component model keeps the core of EL4NET simple, while providing far-going extension potential and giving more power to the developer.

Thanks to this model, the framework is designed to adapt to the architectural changes associated with an application's evolution, making late refactoring cheaper and faster.

Configuration system

The configuration is organized in units of configuration data (the config units). Those units contain the configuration of one or more components. They are contained in freestanding XML files, or are attached to assemblies (dlls, exes) as embedded resources. Config units can depend on, supplement or replace each other. EL4NET builds the complete application configuration out of all those config units; conflicts and dependencies are resolved during this process. To detect misconfiguration already during application start-up, EL4NET performs a structural and semantical validation of the complete configuration when loading it.

The configuration system supports the separation of development related and operation configuration by providing a config token concept. A config token defines a placeholder inside the component configuration. The operation related configuration assigns a value to this placeholder.

The configuration system provides the backbone of the plug-in infrastructure. Config units can be selectively activated or deactivated. An active config unit goes into the whole application configuration, an inactive one does not. Based on this, EL4NET supports plug-ins activated by presence in the application folder. **Advantages:** Thanks to the powerful EL4NET configuration system, the configuration is much easier to maintain. The separation between development and operation related configuration makes it much easier to deploy the application in the production environment. Thanks to the rigorous checks performed at start-up, working with configuration is much simpler and less error prone.

Extended remoting

EL4NET currently allows publishing components using .NET Remoting. The publication system is designed in a flexible way to allow plugging in support for additional technologies such as ASP.NET web services or WCF without changing the framework.

The framework comes with a set of predefined config units for standard publication scenarios, making component publication very easy.

The publication implementation based on .NET Remoting extends the power of .NET Remoting. It supports publishing a component only on a specific set of remoting channels, disallowing access from other channels. Beside the support for traditional remoting channels, EL4NET also allows publishing components on the cross app domain channel. EL4NET already integrates IIOP.NET, a component that allows tightly coupled seamless interoperability with CORBA-based systems (thus including J2EE RMI/IIOP).

Advantages: Thanks to the flexible remoting system, component publication can be chosen at deployment instead of development time.

Flexible deployment

EL4NET supports the use of predefined code compartments to organize your code. For deployment you can flexibly define in what

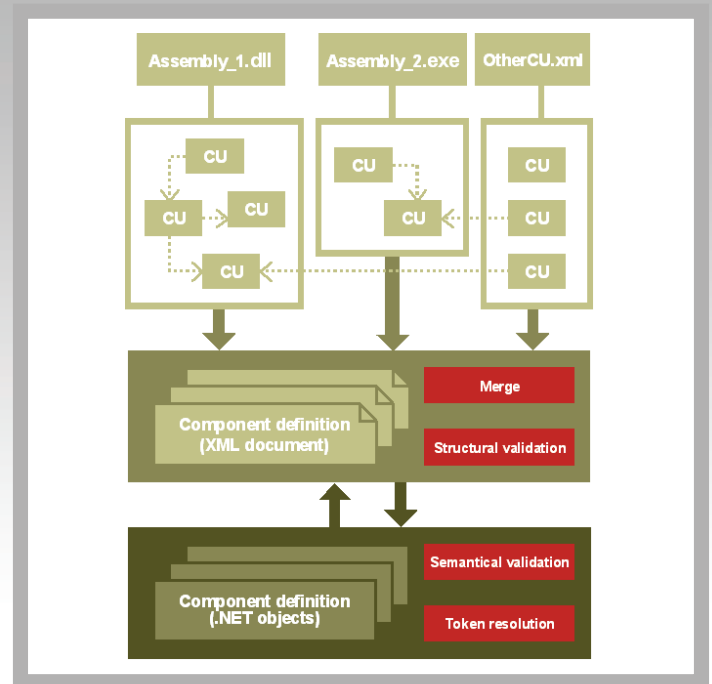


Figure 2: The EL4NET configuration system

process each compartment should run. For example, you can run your 2-compartment-application in 2 appdomains in 1 process (e.g., fat client) or run the same application in 2 separate processes and run it as a thin client and as an application server.

Advantages: EL4NET allows changing the deployment model of the application without requiring any code changes.

More information

Further information can be found at ELCA's website under http://www.elca.ch/Solutions/Technology_Frameworks/EL4NET/EL4NET.php. ELCA's EL4NET is available in both binary and source form. You can obtain the open source project via the sourceforge project page: <http://el4net.sourceforge.net>. Here you can also find a user manual as well as an open discussion forum.

Professional Support in English, French, and German is available. Please contact us via the web site.

With EL4J and IIOP.NET ELCA provides related libraries for other environments as well.

IT-Solutions by ELCA

We make it work.

ELCA

Av. de la Harpe 22-24

case postale 519

1001 Lausanne - Switzerland

Tel. +41 (0) 21 613 21 11

Fax +41 (0) 21 613 21 00

info@elca.ch

www.elca.ch