



POSIX

POSIX Domain for Zeligsoft CX™

Domain specialization is integral to Zeligsoft's approach to model based software engineering and the Zeligsoft CX development tool. Domains capture specific platform knowledge, component framework information and associated validation and transformation engines in order to provide developers with a heightened level of abstraction and measurable productivity gains. Zeligsoft's POSIX (Portable Operating System Interface) Domain was created for developers targeting any POSIX-compliant operating system such as Linux, VxWorks, Integrity and QNX.

Domain-Specific Model Driven Development for POSIX Applications

Through a GUI that presents POSIX concepts, POSIX-specific model elements and the ability to represent the target platform, Zeligsoft CX offers a complete Eclipse environment for the architectural specification and automated generation of the structure and inter-component communication code for a POSIX application. Built-in POSIX concepts such as message queues, threads and processes are exposed by an Eclipse tool palette. As they are used by the developer, they influence how the application is designed and generated.

While domain-specific, the Zeligsoft models also comply with UML 2, ensuring that models and code created in the CX environment can be used with other UML 2-based tools.

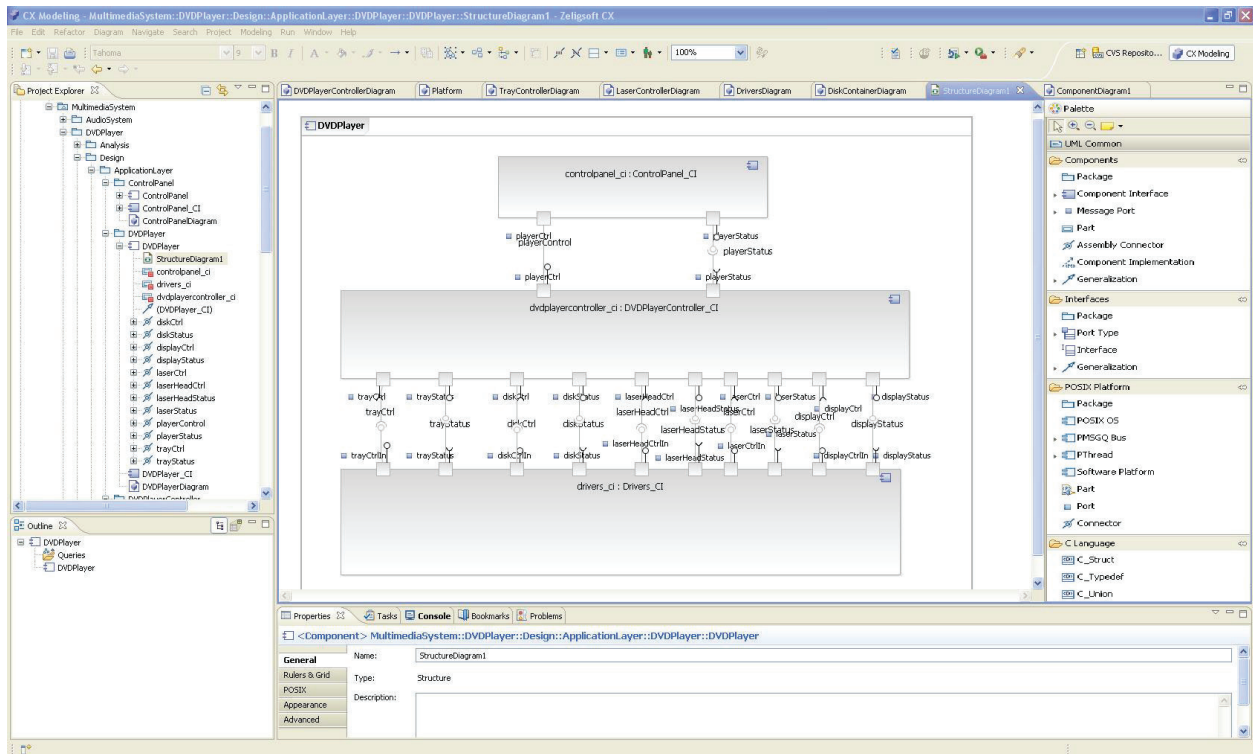
Zeligsoft extends the modeling environment beyond the application to include a model of the target platform and how the application is deployed to this platform. Zeligsoft CX allows developers to create a specific POSIX platform model, in stark contrast to generic tools which rely on an idealized or fixed platform, bringing with it rigid code generation and prescribed runtime code.

Deployment Aware Code Generation

Combining the application and the target platform together within the CX environment is referred to as a deployment. Deployments are a unique Zeligsoft CX capability that enables the application's model elements to be mapped, i.e. deployed, onto the supporting platform elements: Components can be assigned to processes or threads, and Connectors assigned to transports such as POSIX message queues, a CORBA bus or direct function calls. This combined application and platform is then used as the input to Zeligsoft CX's code generation engine. In this way, CX generates deployment-aware code optimized for the target platform. This approach to code generation also frees the designer from having to set up communication mechanisms as this is handled by the generated code.

Runtime entities selected by the designer are incorporated into the generated code; however, only the runtime elements necessary to the application are used making the code base as minimal as possible.

The same POSIX application can be targeted to multiple different software platforms as required. Porting to a new platform or re-partitioning an application requires little more than re-deploying the model elements and then generating structural code that will be optimal to the new deployment.



Zeligsoft CX with POSIX Domain Specialization

Integration

The Zeligsoft development environment completely decouples component behavior and structure. Zeligsoft CX is built on Eclipse and provides integration with the Eclipse CDT (C/C++ Development Tools) Project. When a CDT-based editor is used to develop component behavior, all code is maintained in a single Zeligsoft repository so that manual synchronization between model aspects and behavior aspects is unnecessary. Zeligsoft's build environment compiles and links the component structure and component behavior into a single executable.

Extensibility

Zeligsoft CX can be easily extended to include other domain concepts, for example other inter-component communication mechanisms such as proprietary OS messaging mechanisms or industry-specific standards such as TIPC. By exposing additional busses within the modeling and transformation environment — effectively specializing the domain to the project environment — users benefit from high quality correct-by-construction code and a significant productivity improvement.