# Executable and Translatable UML Summary

Executable and translatable UML (xtUML) accelerates the development of real-time, embedded and technical software systems. Proven and well defined, xtUML is a fully automated methodology utilizing a UML notation.

xtUML is based on an object-oriented approach used on more than 1400 real-time and technical projects. These projects include life-critical implanted medical devices; DOD flight-critical systems; 24x7 performance-critical, fault-tolerant telecom systems; highly resource-constrained consumer electronics; and large-scale, discrete-event simulation systems.

xtUML automation provides many benefits and capabilities including:

- Fully customizable translation that generates 100 percent complete, target-optimized code
- Dramatically reduced defect rates through early execution of target-independent application models and test of application-independent designs
- Accelerated development of products with multiple releases, growing or changing requirements, and families of products
- Concurrent design and application analysis modeling providing compressed project schedules
- Powerful performance tuning and resource optimization
- Effective, practical reuse of target-independent application models
- Effective, practical reuse of application-independent designs
- Rapid project ramp-up resulting from a streamlined UML subset and a well-defined process
- Reduced maintenance costs and extended product lifetimes

xtUML leverages several key concepts that revolve around the complete separation of application and software architecture design:

- Application models capture what the application does in a clear and precise manner. These models are executable, providing the opportunity for early validation of application requirements. Application models are fully independent of design and implementation details.
- Software architectures, defined in terms of design patterns, design rules and implementation technologies, are incorporated into a translator that generates code for the target system. The software architectures are completely independent of the applications they support.
- The translator maps application models to the appropriate design rules and patterns resulting in 100 percent complete code generation for modeled components.

**The xtUML Development Process**

The xtUML development process is concise, robust, and well defined, with clear entry and exit gates. The complete separation of both the software architecture design and the application model provides for not only concurrent design but also application analysis modeling. This concurrency compresses the development lifecycle. Therefore, project schedules are shorter than other processes that sequentially perform the activities.

*Executable UML Models*

An xtUML application model contains the details necessary to both execute and test applications independently of design and implementation. Formal test cases are executed against the model to verify that application requirements have been properly addressed.

No design details or target code need be developed or added for model execution. Application model execution removes system errors early, with less effort and cost, and creates an unmistakably clear exit gate: a completed application model must execute.

**Translation**

xtUML translators automatically generate target-optimized, 100 percent complete code from application models. The translator comprises three pieces:

- A set of design patterns and translation rules (archetypes) that describe the set of patterns to be applied in code generation, when a given pattern should be used, and how xtUML model components will populate or be utilized to build code.
- A translation engine that extracts xtUML application model information, interprets the design patterns and rules, and performs the mapping of model components onto design patterns to generate complete code.
- A run-time library comprising pre-compiled routines that support the generated code modules.

The partitioning of translators into three pieces streamlines the customization, construction, and maintenance of translators. Changes and additions can be made to the design patterns, translation rules, or run-time library without having to contend with the details or code of the translation engine itself.

When generating code, the translator extracts information from the xtUML application model. The translator then selects the appropriate design pattern for the “to-be-translated” model element. The information extracted from this model is then used to “fill in the blanks” of the selected design pattern. The result is a fully coded model component.

This simple approach becomes incredibly powerful for real-life applications. Population of a pattern commonly requires invocation of other patterns and rules. These newly invoked patterns and rules often, in turn, invoke other patterns and rules. The creation of code, for what appears to be one model element, can ultimately involve several nested layers of patterns and rules for multiple model elements. This is fully automated by the translator.

xtUML off-the-shelf translators are available to automatically translate 100 percent of the modeled system to high performance source code that is compact, easily understood, and well documented. Legacy, COTS, hand-written and externally generated code is easily integrated with translator-generated code.

## Performance Optimization

To meet either system performance or constraint requirements, software projects sometimes require multiple pattern implementations for a given model element type. In these situations, a pattern can be defined for each implementation option. The default pattern used for a specific model element can then be overridden to select an optimal pattern for a particular situation. This pattern selection is specified through a process called coloring. Model coloring provides a design engineer with a fine, second level of control over the implementation characteristics and performance of a particular system.

xtUML off-the-shelf translators support a variety of standard, pre-defined coloring options including both performance and size optimization. These open and customizable design patterns can be modified easily to provide powerful performance tuning and resource optimization to support a project's specific needs.

## Integration, Test and Maintenance

xtUML greatly reduces the time and effort required for integration and test by eliminating defects earlier in the lifecycle. Independent application and design testing, elimination of hand-coding errors, and xtUML defect-reduction checkpoints eliminate defects prior to integration and test.

xtUML also simplifies repair of defects and minimizes accidental introduction of new defects into the system. If a design or implementation error is caused by a faulty translator element (pattern or rule), the error will both occur frequently and be quite visible. Such an error is easily traced back to the specific translator element that generated the defect.

Repairing the faulty translator element will quickly propagate the correction throughout all of the system's generated code automatically.

Application model errors result in defective system behavior. Because of the direct mapping between application model elements and the generated code, identification of such a defect leads directly to the specific application model element causing the problem. Fix the defective application model element and the error goes away.

Since a fix is only made at the source of the defect, fixing one problem is unlikely to introduce new problems. The introduction of side effects due to hand-coded modifications is eliminated.

The xtUML approach to system integration, test, and maintenance offers an efficient means for long term support, continued defect fixes, addition of functionality and performance tuning. With xtUML, these activities do not result in increasingly fragile or "brittle" code. Fixes or changes at the application model or software architecture design level always generate clean, concise, well-documented and structured code—without developing the convoluted structure and out-of-sync application analysis, design, and code that can result from code-level modifications.

## Iterative Application Development

With xtUML the complete separation of application from design greatly accelerates and simplifies iterative development. Changing or growing system requirements and system functionality only require modification of the application model. Changes to the application model are quickly and automatically translated to new functionality in the target code, without any modification to the design or manual coding.

## xtUML Application Model Reuse

Separation of design and application enables effective reuse of application models and application partitions (domains). Applications are reusable independently of target characteristics, implementation details, or language. Reuse across multiple products and a product family becomes practical and highly productive.

## xtUML Software Architecture Design Reuse

Separation of design and application also enables effective reuse of software architecture designs. Designs are reusable across multiple systems, independently of the specifics of those systems' applications. Reuse of the same design in multiple applications is not only more productive, but it also guarantees that applications built in different locations, with different developers, will nonetheless smoothly integrate with each other.

## xtUML Benefits

xtUML provides a unique opportunity to both accelerate development and improve the quality, performance and resource utilization of real-time, embedded and technical systems.

xtUML is proven and effective. It has been used for large-scale telecommunication systems with more than four million lines of C++ as well as implanted medical devices with memory measured in Kbytes. It has been used successfully in performance-critical, flight-critical, safety-critical, life-critical, resource-constrained, and cost-constrained systems.

Direct benefits of xtUML include compressed project schedules, reduced software defect rates, optimized system performance and highly compact code. xtUML also accelerates development of products with multiple releases, growing or changing requirements, multiple targets, and families of products. Quality is increased and development accelerated by reuse of pre-tested applications, application components, and designs. Companies can compete more aggressively and effectively for customers, contracts, and market share on a pricing, time-to-market, and product quality basis.

# # #