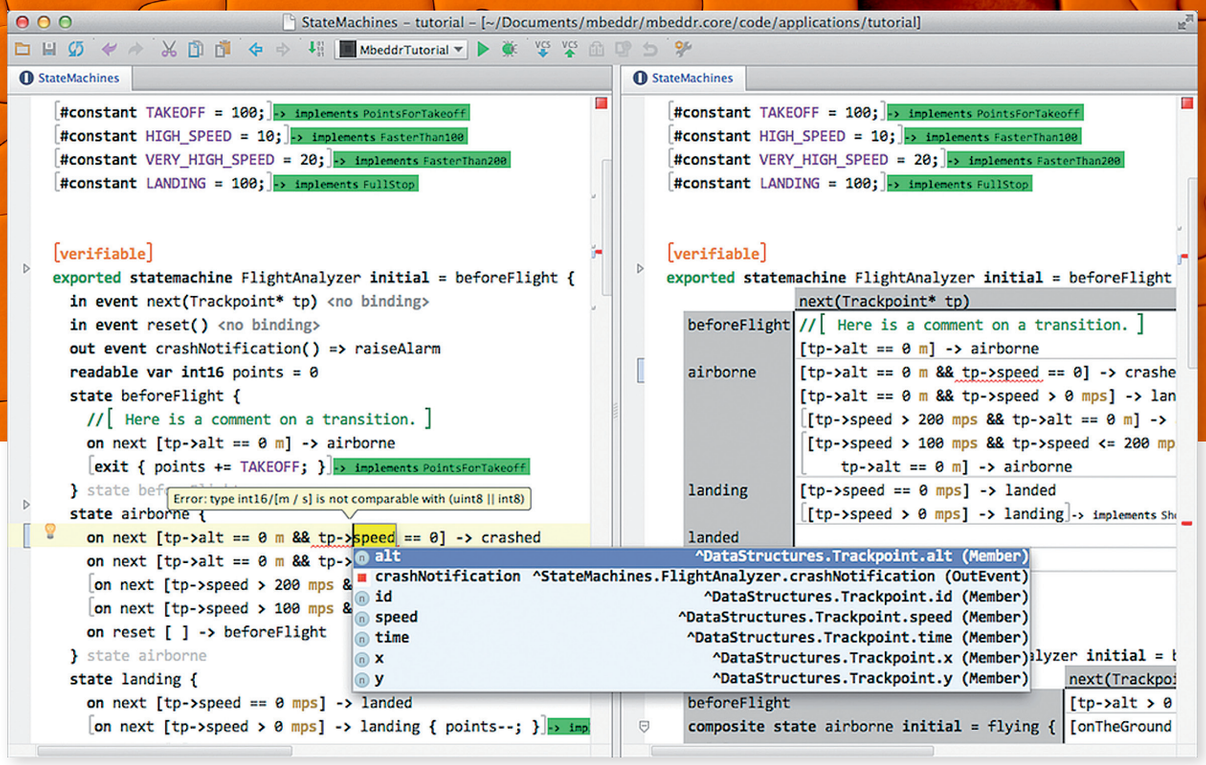
itemis

mbeddr

Engineering the Future of Embedded Software.

Boosting productivity and quality by using extensible DSLs, flexible notations and integrated verification tools.



What is mbeddr?

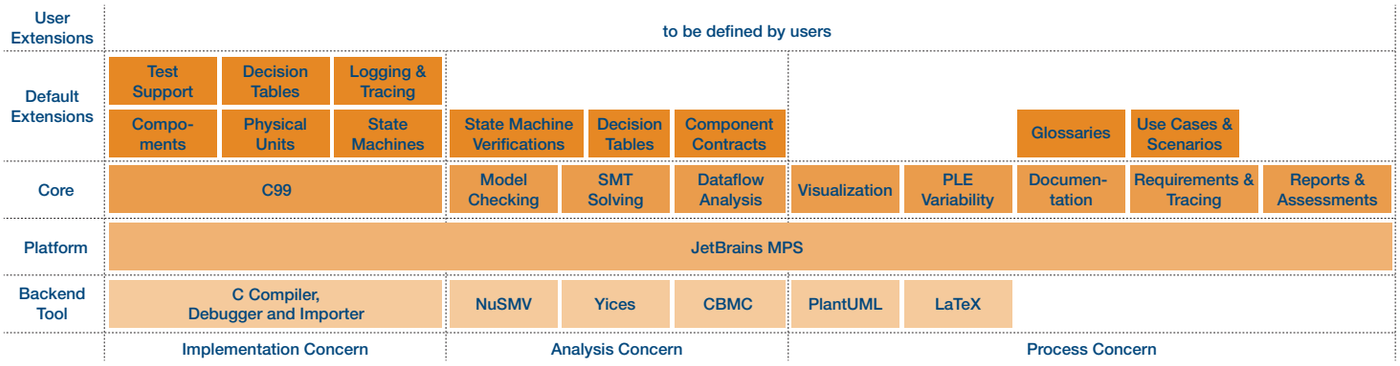
mbeddr is an set of integrated and extensible languages for embedded software development, plus an IDE with refactorings, a debugger and verification tools. It covers all aspects of software engineering, including requirements and product lines, implementation in C as well as C extensions such as state machines, physical units, interfaces and components, testing, and verification. It integrates with build servers. mbeedr is open source software (Eclipse Public License 1.0). It can be found at **http://mbeddr.com**

How is mbeddr extensible?

mbeddr is radically open and supports extension of the tool and of languages. Languages can be extended with new language constructs, typing rules, constraints, generators or IDE features. In addition, arbitrary new languages can be added and (optionally) integrated with existing languages. This is made possible by mbeddr's reliance on JetBrains MPS, a language workbench that supports the definition, composition and use of general purpose and domain-specific languages. It uses a projectional editor, which supports non-textual notations such as tables or mathematical symbols, and it supports essentially unconstrained language composition and extension. Because of the projectional approach, no parser ambiguities can ever result from combining languages. When users build their own extensions, they use the same language engineering facilities that the mbeddr team is using to build the existing languages — user extensions are not second-class.

itemis

mbeddr



Implementation

mbeddr comes with an implementation of C99, with a few minor differences to the standard. The preprocessor is not exposed to the user, first-class concepts are provided for the legitimate uses of the preprocessor (including a module system). mbeddr also comes with C extensions addressing common problems in embedded software development. Interfaces and components support modular software design and reuse. State machines can be embedded in C code. Physical units can be annotated to types in order to make working with physical quantities more robust. Testing, logging and tracing are addressed with first class language constructs. State-of-the-art IDE support is available for all extensions, including syntax highlighting, code completion, real-time type checks and refactorings. The implementation concern also ships with an extensible debugger that is able to debug on the level of the extensions, so the abstractions do not break down when debugging becomes necessary. At the backend layer, mbeddr relies on a C compiler, and a C debugger (gcc/gdb by default).

!

Papers, Tutorials, Screencasts and Download at http://mbeddr.com

Analysis

mbeddr provides static analyses for some of the extensions in the implementation concern. Existing external tools perform the analyses. However, mbeddr integrates the tools tightly by (a) providing language abstractions to conveniently describe behavior that can be analyzed, (b) translating this description to the input of the analysis tool, (c) running the tool, and (d) lifting the output of the tool back to the original abstraction level, to make it easier to understand for the user. The following analyses are available: State machines can be checked with a symbolic model checker, verifying a set of default properties and optional user-defined properties. Decision tables can be checked for completeness and consistency. Feature model configurations are checked for consistency. Finally, interface contracts can be checked statically on components using a C-level model checker.

SYSTEMS BUILT WITH MBEDDR

Smartmeter: mbeddr's components are used in a highly configurable and testable system with more than 100.000 LoC. Physical units improve type safety.

AUTOSAR Software Components: A German car manufacturer is evaluating mbeddr to simplify the development of AUTOSAR software.

A major tool vendor bases their new generation of engineering tools on mbeddr, focussing on legacy code reengineering and controls engineering. Tool to be announced in early 2014.

Process

mbeddr comes with a language for capturing requirements. Traces to requirements can be attached to any program element expressed in any language. Arbitrary additional data, expressed in any language, can be added to requirements. The product line support allows the definition of feature models and configurations. Feature models can be connected to other artifacts by means of presence conditions. While presence conditions are static and work for any language, there is also C-specific support for variability at runtime. The documentation language supports writing prose documents as part of an mbeddr project, exportable as HTML or LaTeX. It supports referencing code (with real references that are renamed if the element itself is renamed) and program code can be embedded as text or as an image. The embedded code is updated whenever the document is regenerated. Visualizations render diagrams from program structures. Reports and assessments are customizable queries over the code.

CONTACT

itemis AG
Kurt Ebert, Director Sales
kurt.ebert@itemis.de

Meitnerstraße 10
70563 Stuttgart | Germany
Tel. +49 711 342191-0
info@itemis.com | www.itemis.com