

















Source code		User interface	
 Lexer	Tokenizer	 Model builder	From scratch
	Model validation		From project
 Parser	Layering	 Wizard	Stage construction
	Parsing		Stage filling
	Validation	 API	Interaction with user
	Product analysis		
 Transpiler	Transpiling		



Source code 	
 Lexer	Tokenizer
	Splits model text into tokens. Token classification rules are defined in cheat sheet.
	Model validation
	Simple validation according to token classification rules. If all tokens were correctly classified, then the model is valid.



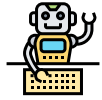
Source code 	
 Parser	Layering
	For providing staged configuration we need firstly to define stages. Layering module detects all dependencies between features and build stage taking into account such dependencies. For example, if one feature requires specific value of another feature, then dependent feature appears only at stage after parent feature definition.
	Parsing
	Parsing token set into an abstract syntax tree: <ul style="list-style-type: none"> • expressions • objects • variables • etc. Product building according to model and user inputs.
	Validation
	Validation of product built.
	Product analysis <p>For dynamic feature modeling we need to have a possibility to modify existing product. To make safe changes we need to provide an product analysis in bundle with its model.</p>




Source code 	
 Transpiler	Transpiling
	Translate model and user fillings to .json format.


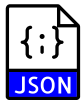
User interface 	
 Model builder	From scratch
	This option proposes to use text field to write a new model. After filling model could be saved and used to build a product.
	From project
	The only difference to previous option is a possibility to analyze a project, find key annotations and build a model skeleton.

User interface 	
 Wizard	Stage construction
	Using parser layering module results builds a wizard stage, placing a set of current layer parameters to define.
	Stage filling
	Defines a type of user input (text field, radio button, checkbox, etc.). After submission, reads user inputs and transmits them to parser.

User interface 	
 API	Interaction with user
	A simple API to interact with user.

User 	
 User	Any alive computer operator.
 System	Any system that has an self optimization mechanism.

Model 	
 Clafer	Any kind of Clafer model.
 Our name	Proposed option of extended Clafer model, that additionally includes complex cardinality intervals and cross-tree constraints with cardinalities.

Product 	
 JSON	Currently only output product supported is JSON model.

Cheat sheet	
Basic Clafer	Could be found at: URL
Extensions	<p>Complex intervals in cardinalities:</p> <div>clafer 1 3 5..7</div> <p>Cross-tree constraints with cardinalities:</p> <div> clafer1 subclafer1 1..3 clafer2 subclafer2 Dependency: subclafer1 Condition: 2 </div>