

Universal LSP and DAP for Modular LWs

Eederico Bruzzone

### Universal Language Server Protocol and Debugger Adapter Protocol for Modular Language Workbenches

#### Federico Bruzzone

Università degli Studi di Milano Computer Science Department PhD Candidate in Computer Science

> 22/07/2024 Cyclus 40th





## Problem Statement Programming Language Implementation

Universal LSP and DAP for Modular LWs

Federico Bruzzone

Problem Statement

Statemen

In a Nutshell

The Reductions of Combination

FOF

LWS

Scientific Contribution

Type System
Components
Modularization

Conclusion

The implementation of a programming language is a complex task that involves several implementation aspects, such as:

- Syntax and semantics definition
- Type system definition
- Code Generation

- Error handling
- IDE support
- Documentation





### Problem Statement Programming Language Implementation

Universal LSP and DAP for Modular LWs

Eederico Bruzzone

The implementation of a programming language is a complex task that involves several implementation aspects, such as:

Syntax and semantics

definition

- Type system definition

- Code generation

- Error handling

- IDF SUPPORT

- Documentation

It is usually done in a monolithic way with a top-down approach, where all the aspects are tightly coupled.





### Problem Statement Programming Language Implementation

Universal LSP and DAP for Modular LWs

Eederico Bruzzone

The implementation of a programming language is a complex task that involves several implementation aspects, such as:

- Syntax semantics and definition
- Type system definition
- Code generation

- Error handling
- IDE support
- Documentation

It is usually done in a monolithic way with a top-down approach, where all the aspects are tightly coupled.

This makes the maintainability, extensibility and reusability of the implementation difficult.





### LSP and DAP In a Nutshell

Universal LSP and DAP for Modular LWs

Federico Bruzzone

Problem Stateme

Stateme

In a Nutshell

The Reduction of Combination

EU5

----

Scientific Contribution

Type System
Components
Modularization

*~* . . .

In 2016, Microsoft in collaboration with Red Hat introduced the Language Server Protocol (LSP) and the Debugger Adapter Protocol (DAP).





### LSP and DAP In a Nutshell

Universal LSP and DAP for Modular LWs

Federico Bruzzone

Problem Statemer

LSP +DA

In a Nutshell

The Reductions of Combination

FOF

- . ....

Scientific Contribution

Components

Conducione

In 2016, Microsoft in collaboration with Red Hat introduced the Language Server Protocol (LSP) and the Debugger Adapter Protocol (DAP).

The LSP and DAP are JSON-RPC Based protocols that allow the communication between a Language Server and an IDE.





### LSP and DAP In a Nutshell

Universal LSP and DAP for Modular LWs

> Federico Bruzzone

Problem Statemer

In a Nutshell

The Reductions of Combination

An Achi

FOF

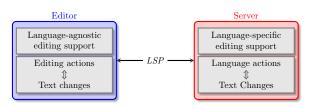
Scientific

Type System Components

Conclusions

In 2016, Microsoft in collaboration with Red Hat introduced the Language Server Protocol (LSP) and the Debugger Adapter Protocol (DAP).

The LSP and DAP are JSON-RPC Based protocols that allow the communication between a Language Server and an IDE.



#### Intrinsic properties:

- Language-agnostic
- IDE-agnostic
- Asynchronous
- Text-Based

#### Features:

- Diagnostics
- Hover
- Go to definition
- Find references





# LSP and DAP The Reduction of Combinations

Universal LSP and DAP for Modular LWs

Federico Bruzzone

Problem Stateme

LSP +DA

LOP TUN

The Reductions

of Combinations

FOF

Scientific

Contribution

Components Modularizatio

Conclusions

Initially implemented for Visual Studio Code, the LSP and DAP have been adopted by several IDEs and programming languages.





## LSP and DAP The Reduction of Combinations

Universal LSP and DAP for Modular LWs

Federico Bruzzone

Problem Statemen

LSP &DAI

The Reductions

of Combinations

FOR

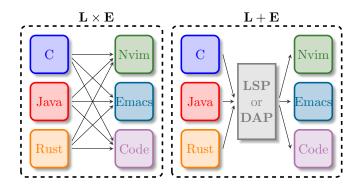
LW

Scientific Contribution

Components

Conclusions

Initially implemented for Visual Studio Code, the LSP and DAP have been adopted by several IDEs and programming languages.







Federico Bruzzone

## LSP and DAP

What would be an important achievement?

Universal LSP and DAP for Servers and IDEs.

Reducing the number of combinations between Language Servers and IDEs.

Problem Statem

1 50 43 45

In a Nicolat at

The O educati

of Combinati

An Achievement

---

- . ....

Scientific Contribution

Components

\_\_\_\_\_

PART.



## LSP and DAP What would be an important achievement?

Servers and IDEs.

Universal LSP and DAP for Modular LWs

> Federico Bruzzone

Problem Stateme

100 40

LSP FDP

The Reduction

of Combination

An Achievement

FOF

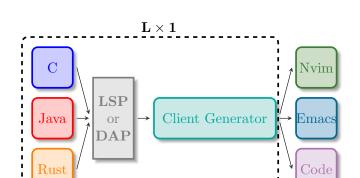
LWs

Scientific Contribution

Components

Modularizatio

Conclusions



Reducing the number of combinations between Language

ROI: Reduce to  $L \times 1$  the number of combinations to support L languages



## Feature-Oriented Programming

Universal LSP and DAP for Modular LWs

Federico Bruzzone

Problem

Stateme

LSP &DAF

The Reduction

of Combination
An Achievemen

FOP

Salamatinia

Scientific Contribution

Components Modularizatio

Conclusion

Feature-Oriented Programming (FOP) is a programming paradigm that allows the development of software product lines (SPLs).





## Feature-Oriented Programming

Universal LSP and DAP for Modular LWs

Federico Bruzzone

Problem Statement

LSP #DAP In a Nutshell

The Reductions
of Combinatio
An Achievemen

FOP

Saiandi Nia

Contribution

Components Modularizatio

Conclusions

Feature-Oriented Programming (FOP) is a programming paradigm that allows the development of software product lines (SPLs).

- Feature is a unit of functionality that satisfies a requirement.
- Feature Model is a model that represents the variability of the SPL.
- Feature Configuration is a set of features that compose a product.

RO 2: Facilitate LSP and DAP Modularization





### Feature-Oriented Programming

Universal LSP and DAP for Modular LWs

> Federico Bruzzone

Problem Statemer

150 47 4

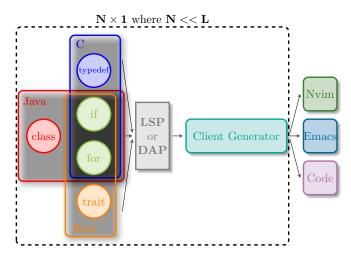
LSP FDA

The Reduction

AN A

Scientific

Components







## Language Workbenches

Universal LSP and DAP for Modular LWs

Federico Bruzzone

Problem Statemer

LSP +DA

The Reductions
of Combinatio

of Combinatio An Achievemen

LWs

Scientific

Type System
Components

Conclusions

Language Workbenches (LWs) are tools that allow the development of programming languages, both GPLs and DSLs.

Language Workbench	Modularization Supp.	Precompiled Feature Supp.	Native IDE gen	LSP & DAP Gen	LSP ≠ DAP Mod.
JustAdd	•	0	0	0	0
Melange	0	0	3rd p.	<b>1</b> /2	À
MontiCore	•	•	•	0	0
MPS	0	0	•	<b>☆</b>	À
Rascal	0	0	•	0	0
Spoofax	0	•	•	<b>☆</b>	À
Ytext	0	•	•	•	0
Neverlang	<b>Ø</b>	•	0	*	☆

- Full support
- No support
- $lackbox{ Limited support}$

- O Coarse-Grained mod.
- My expected contribution
- ☆ Extended contribution

3rd p. Third-party

RO 3: Improve IDE and LSP Generation





## Scientific Contribution

Universal LSP and DAP for Modular LWs

Federico Bruzzone

Problem Stateme

1 50 ¢ DA

CSP TUP

The Reduction

of Combinatio An Achievemen

FOF

Soigntific

Scientific Contribution

Components Modularizatio

Conclusions

- Methodology for whole LWs that support at least component modularization.
- Type System, LSP and DAP Modularization.
- DSL for Type System definition.
- LSP and DAP generation for Neverlang languages.
- Clients and Syntax Highlighting generation reducing the number of combinations.
- Implementation of a Java Library for Neverlang to support the type system, LSP and DAP for every language developed with Neverlang.
- 3 use cases to show the effectiveness of the methodology.

RO 4: Leverage Neverlang for LSP and DAP in LPL Development



Universal LSP and DAP for Modular LWs

> Federico Bruzzone

Problem

1 SD & DA1

In a Nietskel

The Reduction

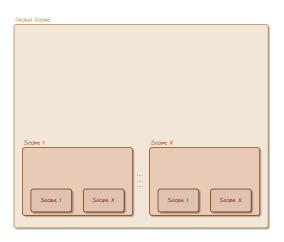
of Combinatio

\_\_\_

LWs

Scientific Contribution

Type System Components



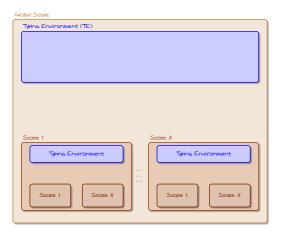




Universal LSP and DAP for Modular LWs

> Federico Bruzzone

Type System Components







Universal LSP and DAP for Modular LWs

> Federico Bruzzone

Problem

LSP +DAF

LSP TUA

The Reduction

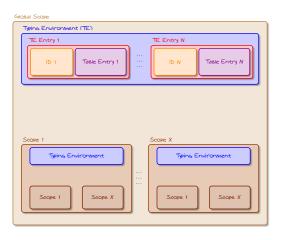
of Combinatio

COS

LWS

Scientific Contributio

Type System Components







Universal LSP and DAP for Modular LWs

Federico Bruzzone

Problem Stateme

1 SD & DA1

LSP FDAI

The Reduction

of Combination

An Achieveme

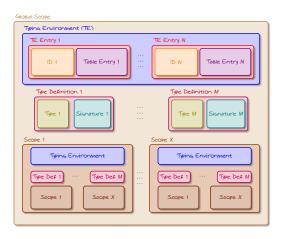
FOR

L 003

Scientific Contribution

Type System
Components
Modularizatio

C----







Universal LSP and DAP for Modular LWs

Federico Bruzzone

Problem

Stateme

LSP +DA

In a Nutshe

of Combinati

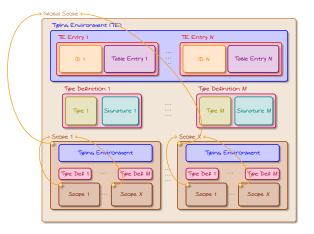
---

LW:

Scientific Contribution

Type System Components

Canalysians







Universal LSP and DAP for Modular LWs

> Federico Bruzzone

Problem Stateme

1 <D &D A1

LSP FDA

The Reduction

of Combinatio

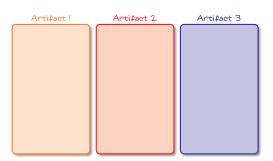
FOR

2003

Scientific Contribution

Type System Components

Modularization







Universal LSP and DAP for Modular LWs

> Federico Bruzzone

Problem Stateme

150 47 45

LSP &DA

The Reduction

of Combinatio

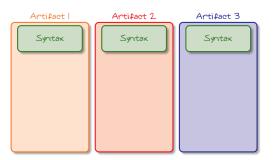
COP

2002

Scientific Contributio

Type System Component

Modularization







Universal LSP and DAP for Modular LWs

> Federico Bruzzone

Problem Stateme

100 40 40

LSP &DAI

The Reduction

of Combinatio

FOR

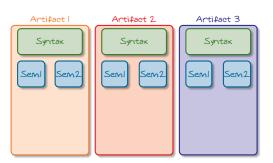
2003

Scientific Contribution

Components

Modularization

Conducione







Universal LSP and DAP for Modular LWs

> Federico Bruzzone

Problem Stateme

LOP TUP

In a Nutsk

The Reductions of Combinatio

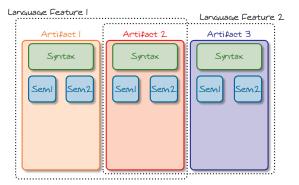
COE

L 00:

Scientific Contribution

Components

Modularization







Universal LSP and DAP for Modular LWs

> Federico Bruzzone

Problem Stateme

LSP +D/

In a Nutsi

The Reduction of Combination

COS

LW:

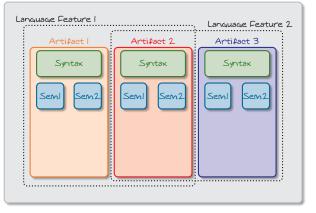
Scientific Contribution

Type System Components

Modularization

Condusions

#### Language Variant







Universal LSP and DAP for Modular LWs

> Federico Bruzzone

Problem Stateme

LSP +DA

In a Nuts

The Reduction of Combination

COE

LW

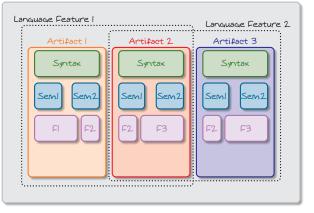
Scientific Contribution

Type System

Modularization

Conclusions

#### Language Variant







Universal LSP and DAP for Modular LWs

> Federico Bruzzone

Problem Stateme

150 474

In a Nutsi

The Reduction of Combination

CO0

LW

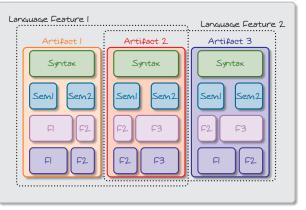
Scientific Contribution

Component:

Modularization

Conclusions

#### Language Variant







Universal LSP and DAP for Modular LWs

> Federico Bruzzone

Problem Stateme

LSP +DA

In a Nutsh

The Reduction of Combination

EQ5

LW:

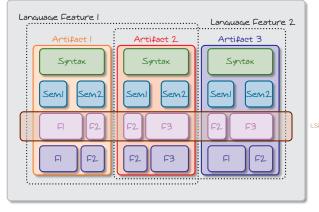
Scientific Contribution

Component

Modularization

Conclusions

#### Language Variant



LSP Variant Feature 1





Universal LSP and DAP for Modular LWs

> Federico Bruzzone

Problem Stateme

In a Nietel

The Reduction

of Combination

FOF

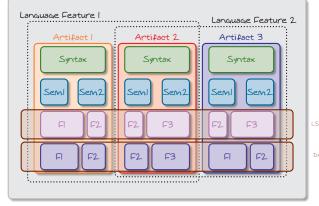
LVV

Scientific Contribution

Modularization

Condusions

#### Language Variant



LSP Variant Feature 1

(Feature 1

DAP Variant Feature 2 Feature 3





## Conclusions Master's Thesis Results

Universal LSP and DAP for Modular LWs

> Federico Bruzzone

Problem Stateme

1 <55 ¢DV.

In a Nutshell

The Reductions of Combinatio

FOF

Saiandi Na

Scientific Contributior

Type System
Components
Modularizatio

Conclusions

- We are writing an article (Code Less to Code More) to be submitted to JSS.

- Propose a feasibility study for the methodology.

- We prototyped the reduction of combinations.

- We prototyped the modularization of the type system.

A TOPE



Universal LSP and DAP for Modular LWs

Federico Bruzzone

Problem Stateme

I SP #DAF

de a Madalan

IN a Nurusiyali

of Commination

of Combination

EU5

. .

Scientific

Type System

Modularia ation

Conclusions

## Thanks for your attention!





## Type Checking and Type Inference

Universal LSP and DAP for Modular LWs

Federico Bruzzone

Problem Stateme

1 <D &D A1

LSP &DA

The Reduction

of Combinatio

FOP

2002

Scientific Contributio

Components

Modularizatio

```
function sum1(x) {
                     1
                    2
                             return sum(x. 1):
                     3
                     5
                         function sum(x, y) {
                    6
                             return x + y;
                     7
                                       Root
                                                              function
             function
               sum1
                                                                sum
parameters
                          body
                                                parameters
                                                                           body
                                        identifier
                                                          identifier
 identifier
                          return
                                                                          return
                                                                         bin_expr
                         call_expr
                                                                identifier
                                                                                  identifier
                identifier
                                 arguments
                  sum
                         identifier
                                            number
```



## Type Checking and Type Inference

Universal LSP and DAP for Modular LWs

> Federico Bruzzone

Problem Stateme

LSP &DAI

The Reductions of Combination

EU5

LW:

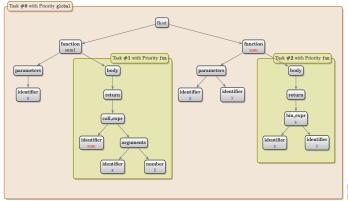
Scientific Contributio

Type System
Components
Modularizatio

```
function sum1(x) {
    return sum(x, 1);
}

function sum(x, y) {
    return x + y;
}
```

- Compilation Unit
- Compilation Unit Task
- Compilation Helper







### Software Product Lines

Universal LSP and DAP for Modular LWs

> Federico Bruzzone

Problem Stateme

LSP +DAF

The Reductions of Combinatio

of Combination

An Achievement

FOP

Scientific

Scientific Contribution

Components Modularizatio

Conclusions

Since 1990s, researchers have been working on the concept of Software Product Lines (SPLs) to move towards a more modular world.





### Software Product Lines

Universal LSP and DAP for Modular LWs

> Federico Bruzzone

Problem Statem

In a Nutshe

The Reductions of Combinatio

EOP

. --

Scientific

Type System
Components
Modularizatio

Conclusions

Since 1990s, researchers have been working on the concept of Software Product Lines (SPLs) to move towards a more modular world.

- SPLs defines a family of software products.
- SPLs is described by a Feature Model.
- A Feature Model describes the variability of the software.
- SPL variants are generated by selecting a set of features.
- A feature (or artifact) is a first-class entity in SPLs.





### Language Product Lines

Universal LSP and DAP for Modular LWs

Federico

Problem Stateme

LSP #DAI

In a Nutshell The Reduction

of Combinatio

FOF

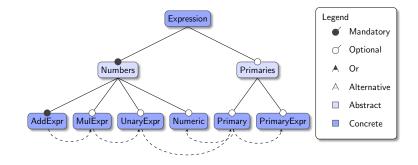
\_ .

Scientific Contribution

Components Modularization

Conclusions

Applying the concept of SPLs to programming languages, we obtain the concept of Language Product Lines (LPLs).







### Language Product Lines

Universal LSP and DAP for Modular LWs

> Federico Bruzzone

Problem Stateme

1 <0 &D4

LSP FDA

The Reduction

An Ad

FOF

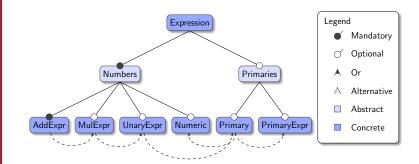
Scientifi

Contribution

Components Modularizatio

Conclusions

Applying the concept of SPLs to programming languages, we obtain the concept of Language Product Lines (LPLs).



## Some achievements:

- Bottom-up approach to language implementation
- Reusability of language artifacts
- Multiple variants of the same language
- Language Workbenches come to the rescue





## Research Objective I

Universal LSP and DAP for Modular LWs

> Federico Bruzzone

Problem Statemer

I SO EDAD

In a Nutshell

The Reductions
of Combinatio

FOF

----

Scientific Contributio

Components Modularizatio

Conclusions

ROI: Reduce to  $L \times 1$  the number of combinations to support L languages

RQ II: How can IDE generation be improved to support LSP and DAP?

RQ 1.2: What are the key challenges in generating LSP and DAP for different programming languages?

RQ 1.3: How can a universal LSP and DAP be developed to support multiple languages and IDEs?





## Research Objective 2

Universal LSP and DAP for Modular LWs

Federico Bruzzone

Problem Stateme

1 <D &DA

In a Nutshel

The Reductions

of Combination

FOF

LW:

Scientific Contributio

Components

Modularizatio

Conclusions

#### RO 2: Facilitate LSP and DAP Modularization

- RQ 21: How can LSP and DAP modularization be facilitated in language workbenches?
- RQ 22: What are the key challenges in modularizing LSP and DAP for different programming languages?
- RQ 23: How can LSP and DAP modularization be integrated with existing language composition and modularization features in language workbenches?





## Research Objective 3

Universal LSP and DAP for Modular LWs

> Federico Bruzzone

Problem Stateme

I SP #DA

In a Nutshell

The Reductions
of Combination
An Arbievemen

FOF

\_ . ....

Contributio

Components Modularizatio

Conclusions

#### RO 3: Improve IDE and LSP Generation

- RQ 31: How can the number of combinations required to support multiple languages be reduced to L × 1?
- RQ 3.2: In what ways does simplifying the development process for language support enhance efficiency?
- RQ 3.3: How does reducing combinations impact the speed and effectiveness of creating language support?





## Research Objective 4

Universal LSP and DAP for Modular LWs

Federico Bruzzone

Problem Stateme

I SP &DAI

LSP &DA

The Reduction

of Combination
An Achievemen

ان

LWS

Scientific Contribution

Components Modularizatio

Conclusions

RO 4: Leverage Neverlang for LSP and DAP in LPL

Development

RQ 4.1: How can Neverlang's LPL development features be leveraged for creating a reusable core for LSP and DAP functionalities?

RQ 4.2: What are the key Benefits of using Neverlang for LSP and DAP development in the context of LPLs?

RQ 4.3: How does leveraging Neverlang's LPL features enhance the scalability and efficiency of LSP and DAP development?





### Journals and Conferences

Universal LSP and DAP for Modular LWs

> Eederico Bruzzone

Conclusions

#### Journals

- JSS (Journal of Systems and Software)
- TSE (IFFE Transactions on Software Engineering)
- TOSEM (ACM Transactions on Software Engineering and Methodology)
- TOPLAS (ACM Transactions on Programming Languages and Systems)

#### Conferences

- ICSE (International Conference on Software Engineering)
- PLDI (Programming Language Design and Implementation)
- OOPSLA (Object-Oriented Programming, Systems, Languages, and Applications)
- SLE (Software Language Engineering)





## Language Workbenches and Research Groups

Universal LSP and DAP for Modular LWs

> Federico Bruzzone

Problem Stateme

150474

In a Nutshell The Reductions

The Reductions of Combination An Achievement

FOF

Scientific

Contributio

Components Modularizatio

- JustAdd  $\rightarrow$  Computer Science department of the Lund University (Lund, Sweden)
- Melange → DiverSE research team at the Institut National de Recherche en Informatique et en Automatique (INRIA) (Paris, France)
- $MontiCore \rightarrow Software$  Engineering group at the RWTH Aachen University (Aquisgrana, Germany)
- MPS → JetBrains Research (Saint Petersburg, Russia)
- Rascal → Centrum Wiskunde & Informatica (CWI) (Amsterdam, Netherlands)
- Spoofax → Delft University of Technology (Delft, Netherlands)
- Xtext → Edipse Foundation (Ottawa, Canada)





### Language Workbenches and Research Groups

Universal LSP and DAP for Modular I Ws

> Federico Bruzzone

Problem Stateme

LSP +DA

The Reductions
of Combination

FOF

\_\_\_\_

Scientific Contributio

Components

Modularizatio

Condusions

- Martin Fowler: Renowned for his work on software development methodologies.
   His Book "Domain-Specific Languages" is a seminal work in the field.
- Markus Voelter: Known for his contributions to the development and promotion of language workbenches like JetBrains MPS.
- Eelco Visser: A professor at Delft University of Technology, Visser has made significant contributions to the field through his work on the Spoofax language workbench.
- Gregor Kiczales: Known for his work on aspect-oriented rogramming (AOP).
   Professor at the University of British Columbia.
- Antonia Bertolino: known for her work on software testing and quality assurance.

