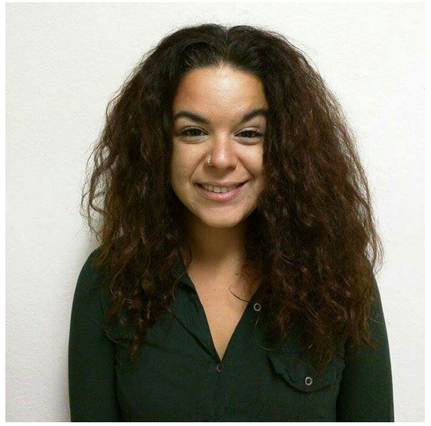
Experience from ERC-Starting



Niki Vazou
IMDEA Software Institute
Madrid, Spain

Hey, I am Niki Vazou!

2005-2011	Diploma	NTUA, Greece
2011-2017	PhD	UCSD, USA
2017-2018	Post-doc	UMD, USA
2018-now	Research Ass. Prof.	IMDEA, Spain

ERC Starting

The proposal

The interview

ERC Starting

The proposal

The idea

B1 (5 pages)

B2 (14 pages)

The idea:

Something that has impact and only you can do.

Only you: explicitly justify it!

Impact: foundational,
not completely realistic,
but gradual.

B1 (5 pages)

1: Motivation & Goal

2: Objectives & Methodology

3: Risk & Difficulties

4: Potential Impact

B2 (14 pages)

1: Introduction

2: State of the Art

3: Objectives

4: Methodology

5: Conclusion

B1 is the summary of B2!

B1

1: Motivation & Goal

Start with a concrete example!

Section a: Extended Synopsis of the scientific proposal

a.1 Motivation and Goal

Software verification validates that programs satisfy certain properties to ensure the absence of critical software bugs. The more costly software bugs become, in terms of money or debugging time, the more willing software developers are to ensure their absence using software verification techniques.

Refinement types [18] are a modern software verification technique that extends types of an existing programming language with logical predicates, to verify critical program properties not expressible by the existing type system. For example, consider the function `get xs i` that returns the `i`th element of the list `xs`. The existing type below states that `get` takes a list of `a`s, an integer and returns an `a`.

Existing Type: `get :: [a] → Int → a`
Refinement Type: `get :: xs:[a] → i:{Int | 0 ≤ i < len xs} → a`

The type of `get` gets *refined* to enforce in-bound indexing, a property that the existing type system cannot encode. Concretely, the refinement `0 ≤ i < len xs` on the index `i` requires that `get` can only be called with indices in the bounds of the input list. Such assertions are checked statically and can be used to prevent critical, real-world bugs, e.g., the memory violation of the infamous Heartbleed bug, without the need of runtime checks that increase a program's execution time.

B1

1: Motivation & Goal

Finish with a concrete goal

program developers that put extra effort and use idiomatic code to optimize program's runtimes.

The goal of this proposal is to develop a both practical and sound verification system.

a.2 Objectives and Methodology

Certified refinement types (CRETE) will construct *sound* proofs for software verified by SMT-automated, *practical* refinement types. We will define explicit certificates that capture the SMT proofs and use them to derive Coq and HOL-style proofs of the original software. CRETE will be used to verify real world code, such as cryptographic protocols and secure web applications.

Figure 1 summarizes the workflow and scientific objectives of CRETE. The user will interact with SMT-automated refinement types to verify code developed in an existing programming language. The first objective of CRETE is to mechanically annotate the programs accepted by the refinement type checker with explicit certificates that will be independently validated or tested against the program's runtime semantics; thus addressing *SoU1*, i.e., the discrepancies between program and SMT semantics.

B1

2: Objectives & Methodology

THE BOX

program developers that put extra effort and use idiomatic code to optimize program's runtimes.

The goal of this proposal is to develop a both practical and sound verification system.

a.2 Objectives and Methodology

Certified refinement types (CRETE) will construct *sound* proofs for software verified by SMT-automated, *practical* refinement types. We will define explicit certificates that capture the SMT proofs and use them to derive Coq and HOL-style proofs of the original software. CRETE will be used to verify real world code, such as cryptographic protocols and secure web applications.

Figure 1 summarizes the workflow and scientific objectives of CRETE. The user will interact with SMT-automated refinement types to verify code developed in an existing programming language. The first objective of CRETE is to mechanically annotate the programs accepted by the refinement type checker with explicit certificates that will be independently validated or tested against the program's runtime semantics; thus addressing *SoU1*, i.e., the discrepancies between program and SMT semantics.

B1

3: Risks and Difficulties

Your chance to defend the weaknesses of the proposal!

a.3 The risks and difficulties

The CRETE project is high-risk because (1) it proposes a derivation of Coq proofs from real programs and (2) it proposes the design of a logic for refined System F_ω. To ameliorate risk (1) we will carefully engineer the proposed translation and we will impose the necessary restrictions on the original system. To ameliorate the risk (2) we can restrict the system to a calculus that is sound but expressive enough to address the practical applications studied in CRETE. The PI is an active developer of Liquid Haskell for 9 years and has extensive experience with the design and implementation of practical refinement types, thus she is in the unique position to distinguish the features of refinement type systems that are critical for real world applications. Further, the PI has active collaborations with experts in the areas of type theory and program semantics, e.g., Michael Greenberg, Alex Kavvos, and Gilles Barthe. Finally, IMDEA, without any teaching obligations and with expert colleagues (in particular Aleksander Nanevski), provides a perfect work environment to carry out such a risky experiment.

B1

4: Potential Impact

Be **BOLD**

a.4 Potential Impact

Scientific Impact: CRETE aims to develop the principles of refinement types and crystalize their connection with type theory. This deeper understanding will further encourage the adoption of refinement types systems by the verification and programming languages communities and shed new light to important, open research problems for example, error reporting and inference of type specifications.

Socioeconomic Impact: Refinement types have already been used in industrial software. Yet, the soundness problems are known and discourage further adoption. CRETE, via the current front-end of automated, practical refinement types, makes sound verification accessible to industrial developers. This low-cost, high-profit approach will encourage further adoption of formal verification and potentially refinement types will, by design, be integrated in future mainstream programming languages.

Educational Impact: The results of this project can be used for educational purposes. Refinement types have been taught in advanced undergraduate and graduate classrooms. This project aims to make software development supported by formal verification so attractive, that can be used as an aid on programming courses and promote verified programming as the de facto way to teach programming.

B2: “Just” expansion of B1

Advice: Write a B1, so strong that nobody looks at B2.

ERC Starting

The proposal

The idea

B1 (5 pages)

B2 (14 pages)

ERC Starting

The proposal

The interview

Presentation (10 min)
Questions (15 min)

The interview

Questions (15 min)

What: From Panel & Expert Reviewers.
Goal: Get Panel Questions.

Preparation: Ask expert collaborations for Qs.
Advice: Be confident!

The interview

Questions (15 min)

Q: Can you apply it to X?

A: Great idea, not in my plan. Y might be interested.

The interview

Questions (15 min)

Q: You only address functions, what about hashes?

A: But everything is a function! There is a translation.

The interview

Presentation (10 min)

1: Goal is to engage the panel.

2: The panel is very smart.

3: The panel has read your B1, and liked it.

Q: Should the presentation be technical?

Q: Should the presentation be technical?

version i: Very technical.

MOCK: ☹ **COACH:** ☺

version j: High level.

MOCK: ☺ **COACH:** ☹

final: Quite technical.

MOCK: ☹ **ERC:** ☺

The interview

Presentation (10 min)
Questions (15 min)

Personally, very stressful:

Too much time spend for 10 minutes!

“You develop skills to explain concepts to students.”
— Aleks Nanevski

ERC Starting

“Paper writing is so much easier after ERC!”
— Niki Vazou (me)

“You develop skills to explain concepts to students.”
— Aleks Nanevski

Thanks!

ERC Starting

The proposal

The idea

B1 (5 pages)

B2 (14 pages)

The interview

Presentation (10 min)

Questions (15 min)