Hum* 5

Bill J. Ellis
February 9, 2005

iSpeak @ 1^{st} SE Co-Ex @ Px-HIS

Contents

*Hums are short notes intended for distribution between those involved with EPSRC grant GR/T12289/01. Hums describe ϵ-baked ideas, where $1 \geq \epsilon \geq 0$. (Hum refers to both the Praxis Humming bird and Winnie-the-Pooh's indirect approach to writing: "Poetry and Hums aren't things which you get, they're things which get you.").

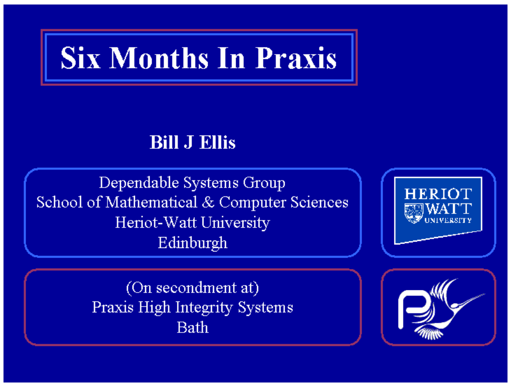
1 Introduction

I was invited to give a presentation at the first meeting of the Software Engineering Community of Expertise within Praxis High Integrity Systems (SE Co-Ex @ Px-HIS). Being new, the exact format and goals of the community have yet to be established. However, the fundamental idea is to create a formal community for those within Praxis wanting to learn (and promote) best practise in Software Engineering. It is likely that members of the community will present at 'show and tell' seminars, to keep others informed of emerging ideas and technologies they have encountered. Further, an informal newsgroup encouraging the dissemination of Software Engineering tit-bits, questions, observations, etc. will probably appear on the Praxis intranet.

My presentation summarised the research that led to this Praxis secondment and the work that will be undertaken. This note collects the slides, bullet point notes, and question and answers associated with this presentation.

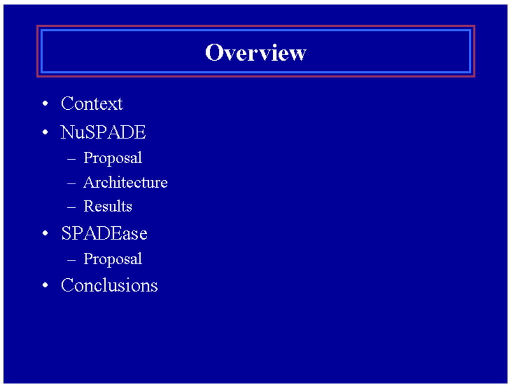
2 Presentation

2.1 Six months in Praxis



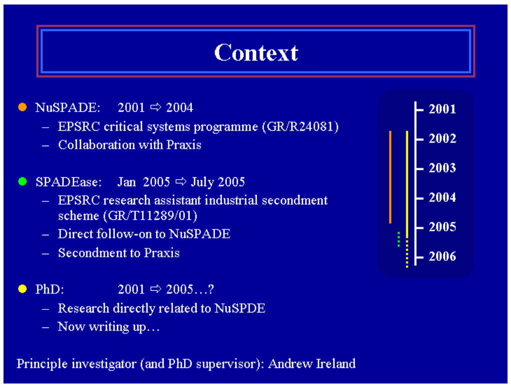
- I am Bill Ellis.
- I am a research associate and PhD student at Heriot-Watt University.
- Currently on secondment at Praxis.
- Short talk about why I am here and what I will be doing.

2.2 Overview



- Begin by explaining the context of this secondment.
- Give a very brief summery of the previous project I was involved with at Heriot-Watt, called NuSPADE.
- Outline this new project at Praxis, called SPADEase.
  - Which is a direct follow-on to NuSPADE.
- Finish with some some conclusions.

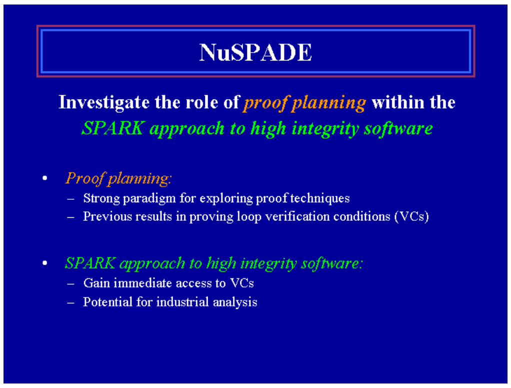
2.3 Context



- The NuSPADE project ran for three years.
  - Funded under the EPSRC[1] critical systems program.
  - A collaboration with Praxis.
  - Principal investigator: Andrew Ireland.
  - Research associate: Myself.
- SPADEase is the project name for this six month secondment at Praxis.
  - Funded through the EPSRC research assistant industrial secondment scheme.
  - This scheme is applicable.
    - To recently completed EPSRC projects.
    - Which had a strong and continuous industrial collaboration.
  - NuSPADE is a perfect example of this kind of project.
    - We applied and we got finding.
  - Second the research associate from the recently completed EPSRC project to the industrial collaborator.
    - To facilitate technology transfer.
      - Transfer ideas from the original research project into industry.
    - To improve the training of the research associate.
- I am also a PhD student.
  - My research is directly related to the NuSPADE project.
  - Am now writing up.

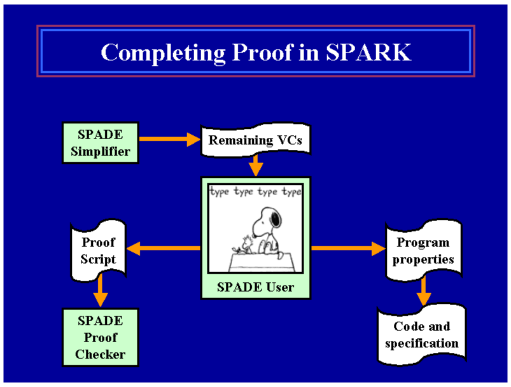
[1]Engineering and Physical Sciences Research Council (EPSRC)

2.4 NuSPADE



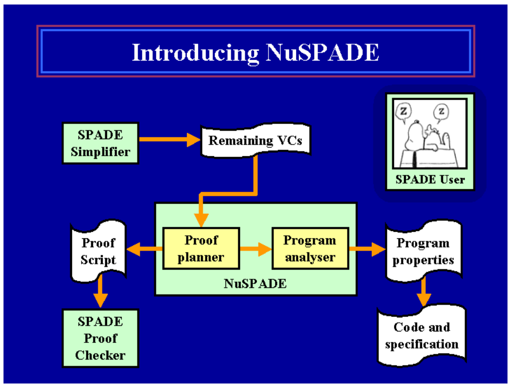
- The original NuSPADE project title.
  - Investigate the role of proof planning within the SPARK approach to hight integrity software.
- Proof planning was selected at the outset.
  - Is a strong paradigm for exploring proof techniques.
  - Another Student of Andrew Ireland, Jamie Stark, had produced encouraging results in applying proof planning to loop based VCs.
- SPARK was selected because.
  - Its tools could be used to gain immediate access to VCs.
    * No need to create formal programming language (As SPARK).
    * No need to build a VC generator (As Examiner).
  - SPARK is actually used in industry.
    * Potential for evaluation on real industrial examples.

2.5 Completing proof in SPARK



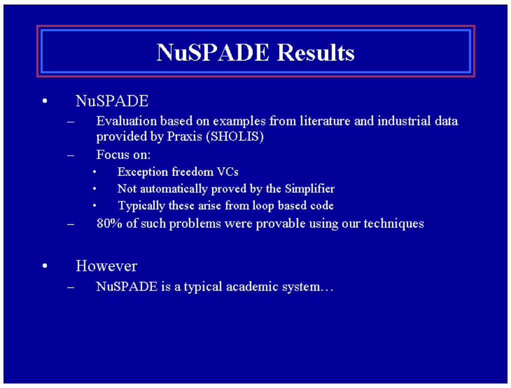
- A brief reminder of of how proof is completed within the SPARK approach.
- The Simplifier automatically proves around 95% of execption freedom VCs.
- But action is required on the remaining 5%.
- The user inspects the remaining VCs.
  - Provable VCs are proved interactively using the Proof Checker.
  - Unprovable VCs indicate a problem in the code or specification.
    * This may be addressed by adding additional program properties to the code.

2.6 Introducing NuSPADE



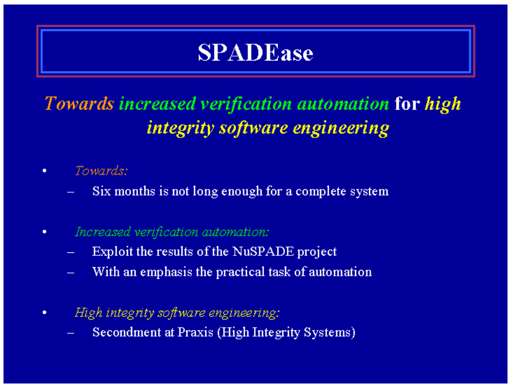
- NuSPADE aims to automate the role of the user.
  - It achieves this through an integration between proof planning and program analysis.
- The remaining VCs are tackled by a proof planner.
  - If a proof is found.
    * A proof script is generated and checked in the Proof Checker.
  - If a proof is not found.
    * The failure pattern may indicate the need for stronger program properties.
    * In this case the proof planner seeks the services of a program analyser.
    * If the program analyser finds these properties, the code is annotated accordingly.

2.7 NuSPADE results



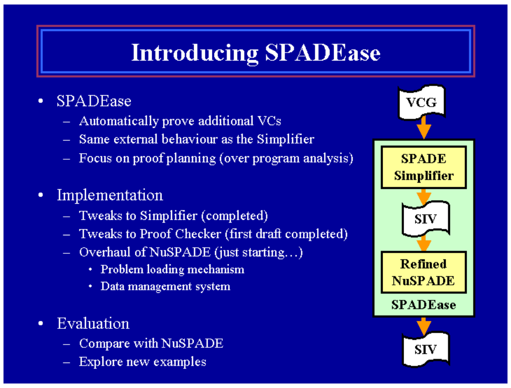
- Evaluation based on.
  - Examples from literature.
  - Industrial data provided by Praxis (SHOLIS).
- Focus on the exception freedom problems.
  - Not automatically proved by the Simplifier.
  - Typically these arise from loop based code.
- We found that 80% of such problems were provable using our techniques.
- However.
  - NuSPADE is a typical academic system.
  - It is far from a commercial product.

2.8 SPADEase



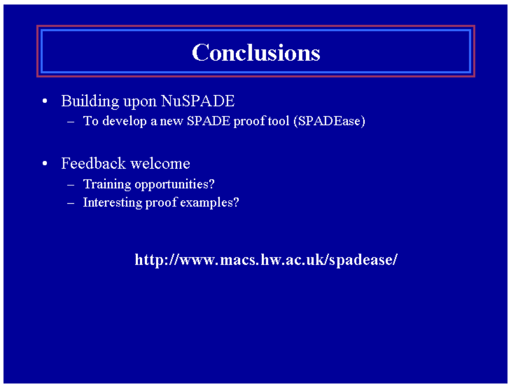
- The original SPADEase project title.
  - Towards increased verification automation for high integrity software engineering.
- Towards.
  - Highlights that six months is too short for a complete system.
  - Instead is more about demonstrating the viability of a complete system.
- Increased verification automation.
  - Exploit the NuSPADE project with an emphasis on automation.
- High integrity software engineering.
  - Work will take place within Praxis.

2.9 Introducing SPADEase



- SPADEase aims to automatically prove additional VCs.
- It will have the same external behaviour as the Simplifier.
  - It will be implemented as a wrapper around the Simplifier.
  - Containing a refined version of NuSPADE.
- Due to time constraints, will focus on the proof planning side.
  - However, may still explore some light-weight program analysis features.
- In terms of implementation.
  - Need some tiny changes to the Simplifier for integration purposes.
    * Which have been completed. And checked in.
  - Need some larger changes to the Proof Checker so it can be controlled by SPADEase.
    * A first draft has been completed.
    * To be finalised once SPADEase is operational.
  - Require an overhaul of NuSPADE.
    * This is just starting, and will probably take a couple of months.
- For Evaluation.
  - Initially compare with NuSPADE.
  - And hopefully explore some new examples.

2.10 Conclusions



- Building upon NuSPADE to develop a new SPADE proof tool called SPADEase.
- Feedback is welcome.
  - Do you have any interesting proof problems?
  - Or know of relevant training opportunities?
- Can follow the project on the project web page.
  - http://www.macs.hw.ac.uk/spadease/

3 Questions and answers

- **Q** - SHOLIS is an older system, lacking quantified annotations. The SPADE Simplifier may fair better with the quantified annotations seen in modern SPARK systems.
- **A** - The idea behind NuSPADE and its evaluation is that it tries to automate the proof of unannotated code - discovering and inserting invariants accordingly. Thus any user annotations (quantified or otherwise) would be striped prior to running NuSPADE.

- **Q** - How would NuSPADE cope with partial correctness problems?
- **A** - The architecture of NuSPADE could support automated partial correctness proofs. However, as we target exception freedom, the current heuristics within NuSPADE would be very poor indeed.

- **Q** - What is the difference between SPADEase and the SPADE Simplifier?
- **A** - The SPADE Simplifier does not separate proof search from proof checking. More flexible heuristics can be implemented during proof search if not having to simultaneously ensure soundness.

- **Q** - So, could the SPADE Simplifier be adapted to this alternative style?
- **A** - Yes. A rational reconstruction of the SPADE Simplifier, with respect to the proof planning paradigm, would be possible. However this would be a difficult exercise (certainly beyond six months).