SPARK Update - November 2004

Rod Chapman

Praxis High Integrity Systems

Contents

- What have we been doing for the last few years?
- What's coming next?
- Future plans.
- An appeal…

The last few years

- Many, Many things…far too many to mention in detail:
  - RavenSPARK
  - Tool and partner integrations
  - New projects
  - The new SPARK book
  - New "Black Belt SPARK" Course
  - SEI, PSP, and SPARK
  - Security
  - Exception Freedom and Theorem Proving improvements

RavenSPARK

- Tasking is back!
- Brings a subset of the Ada95 Ravenscar Profile directly into the core of SPARK.
- Deterministic scheduling scheme, suitable for hard-real time schedulability analysis
- RavenSPARK eliminates potential errors and much more…

Tool Integrations and Partners

- Tool vendors now supporting SPARK…
  - ARTiSAN Real-Time Studio
  - Ilogix Rhapsody-in-Ada
  - ADI Beacon
  - High Integrity Solutions VDS
  - Plus significant support from compiler vendors
  - More to come…
- Marketing/Sales/Training partnership in the USA with Pyrrhus Software

Some recent projects

- The MULTOS CA
  - A distributed, highly secure system.
  - Designed for long life and extensibility.
    - e.g. new cryptographic algorithms
    - Significant extensions implemented successfully during development
  - REVEAL requirements.
  - INFORMED high-level design to 9-processor distributed system.
    - Programming languages: C++ (auto generated), Ada95, SPARK, C, SQL!
  - Whole project adopted and maintained by Mondex now.

Some recent projects (2)

- Aircraft Test Equipment
  - SIL0, but SPARK anyway.
  - UML using Rational Rose.
  - Better *and* cheaper!
- ULTRA SAWCS
  - INFORMED, SPARK, UML using ARTiSAN Real-Time Studio.
- AerMacchi
  - M346 Jet Trainer flight control system. INFORMED, SPARK, MATLAB/Simulink.
- ARINC
  - Airborne health monitoring system.
- NATO C3 Agency
  - Secure information downgrader

The new SPARK Book

- High Integrity Software: the SPARK Approach to Safety and Security by John Barnes and Praxis
- Published in April 2003.
- Good reviews on SlashDot, Amazon, comp.risks, ACM Computing Surveys.
- Has generated much "buzz"

"Black Belt SPARK" course

- New, advanced course for those with experience of using SPARK.
- Focus on how to make best use of the proof facilities, and in particular the proof of the absence of exceptions.
- "Proof Directed Software Design" - how do you write *provable* code?

SEI, PSP and SPARK…

- SEI have discovered Correctness-by-Construction and SPARK…
- Praxis have discovered PSP/TSP.
- SPARK projects can deliver <= 0.1 defects per kloc.
- So can PSP/TSP projects…
- What happens if you put the two together?

Security

- *Finally*, the security community are taking high-integrity software very seriously.
- SEI/DHS report on software development for secure systems
  - Only 3 processes identified that can deliver fewer than 0.1 defects per kloc: TSP, IBM CleanRoom and Praxis CbyC.
- SPARK has been noticed by the NSA, GCHQ, and even Microsoft (!).

Exception Freedom and Theorem Proving

- A SPARK program can be shown to be free of all "predefined exceptions"
  - e.g. buffer overflow, division by zero, range violation etc.
- We do this by generating small **conjectures** from a program, the proof of which show that the exception could **never** occur,
  - Good news - Proof process is automated by the **Simplifier** - a theorem prover.

Exception freedom

- Exception freedom proof - why is it important?
  - Can be attempted without a formal spec., or explicit pre- and post-conditions, so is approachable.
  - Provides *evidence* that compiler-generated checks can be turned off with justification.
  - This implies significant simplification of generated code and subsequent coverage analysis. Saves money.
  - Forces you to really *think* about your code. Finds bugs.
- You mainly need CPU cycles for theorem proving - and these are cheap.

The impact

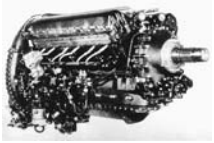
- Require simplified proofs **prior** to code review. Why not? "We have the technology…"
- Many classes of defect are simply impossible.
- Correctness emerges as a side-effect of having had to think about it!
- Code review can then focus on more important matters - e.g. safety and security.
- You can **justifiably** compile "with checks off". Simplifies coverage analysis.
- Integration and "first target run" should be easy.

Turning the dials up…

- Three ways to improve performance…

– Smarter VC Generation and Theorem Proving Tactics

– Streamline existing tactics and algorithms



– Get a bigger engine…

The Test Data…

- "Project R"
  - Embedded, real-time stores-management system
  - Some functions are SIL3
  - 22968 declarations and statements

The Test Data…

- "Project R" Verification Conditions

VC Class	
Assertion or Postcondition	7142
Precondition	69
Exception freedom	10890
Refinement	554
Inheritance	0
Total	18655

Test Conditions

- Which combination of Examiner/Simplifier/Hardware to use?
- Principle: use tools and hardware that's available to users.
  - Commercial releases of tools
  - Commodity PC hardware
- Measure:
  - Execution time of tools
  - Simplifier "hit rate"
  - Number of VCs left to be reviewed or proven manually.

Performance data

Toolset	Hardware	Time/mins	Hit rate %	VCs left
6.3	1.8GHz P4 Mobile	111	94.5	1025
7.0	1.8GHz P4 Mobile	109	94.69	990
7.0	2.4 GHz P4 Xeon	73	94.69	990
7.1	2 * 2.4 GHz P4 Xeon	49	95.75	791
Wavefront (Examiner 7.2, Simplifier 2.16rc3)	2 * 2.4 GHz P4 Xeon	82	97.24	515

What's coming next?
Release 7.2 - Coming soon

- Examiner
  - Handling of file-name cases normalized on Windows and Solaris.
  - /brief switch - produces "gcc style" error messages for IDE integration.
  - Correct Flow Analysis of static-range "for" loops.
  - VCG assumes R-Values of local variables "in type" when /rtc or /exp used.
  - Correct VCG modelling of "for" loops with a dynamic range.
  - Proof involving completion of private types.
  - Generation of proof rules for composite constants.

Release 7.2 - Coming soon

- Simplifier 2.15 (done)
  - New proof tactics for dealing with "for_all" conclusions and nested composite object updates.
  - Significant improvement in VC proof for RTCs involving arrays and records.
- Simplifier 2.16 (available soon)
  - More tactics: arithmetic (particularly mod, div, **)
- Simplifier 2.17+
  - logic, and use of proof rules for composite constants.

SPARK: Future Plans (1)

- Parallel Simplification!
- All theorems are completely independent, so if one £1500 laptop PC can do 70 theorems per minute, why not use a network of N PCs?
- We are now working on a distributed, client-server Simplifier.
- Simplifier is CPU-bound, so near linear speedup is attainable.
- How many (mostly unused) fast PCs have you got lying round the office?

SPARK: Future Plans (2)

- Expand language subset:
  - Tagged Types ("OO stuff") - DONE
  - Ravenscar tasking profile - DONE
  - Generic units - Design in progress

An appeal…

- How come you never call???
- You pay for our maintenance and support service…Please use it!
- Email: sparkinfo@praxis-his.com
- Phone: +44 1225 823829
- Web: www.sparkada.com

Conclusions

- SPARK continues to grow in size, maturity and use.
- A marked change in attitude has been observed:
  - Tool vendors are coming to see us…
  - People have read about SPARK and are interested enough to come and find us at shows…
  - The book…
  - The security community…