VDM++ as a Basis of Scalable Agile Formal Software Development

Hiroshi Mochio[1] and Keijiro Araki[2]

[1]Chikushi Jogakuen University (mochio@chikushi-u.ac.jp)

[2]Kyushu University (araki@csce.kyushu-u.ac.jp)

Outline

- Background
- Agile Method and Its Problems
- Scalable Agile Method
- Formal Method
- Scalable Agile Formal Method (SAF Method)
- Three Key Points for SAF with VDM++
- Conclusion

An (Bitter) Experience

- Some University (about 3,500 students)
- Planning to Build a New SIS
- Now Getting Requirements

In such case, if you have to participate in the development process ….

- ➢ Too many sudden shifts in the requirements.
- ➢ A messy confusion.
- ➢ When to be terminated?

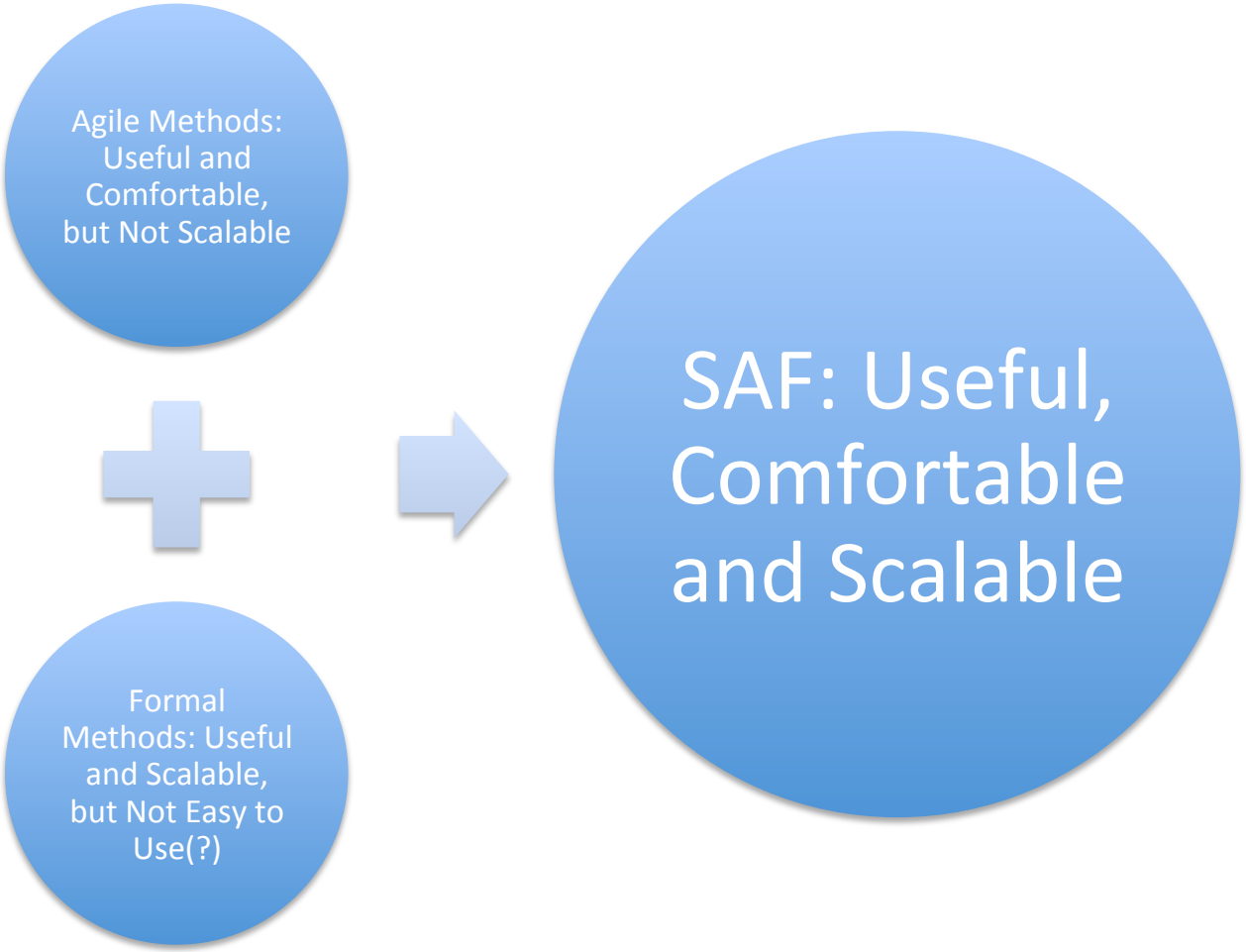
The Truth

- In building software requirements changes are the norm, the question is what we do about it.

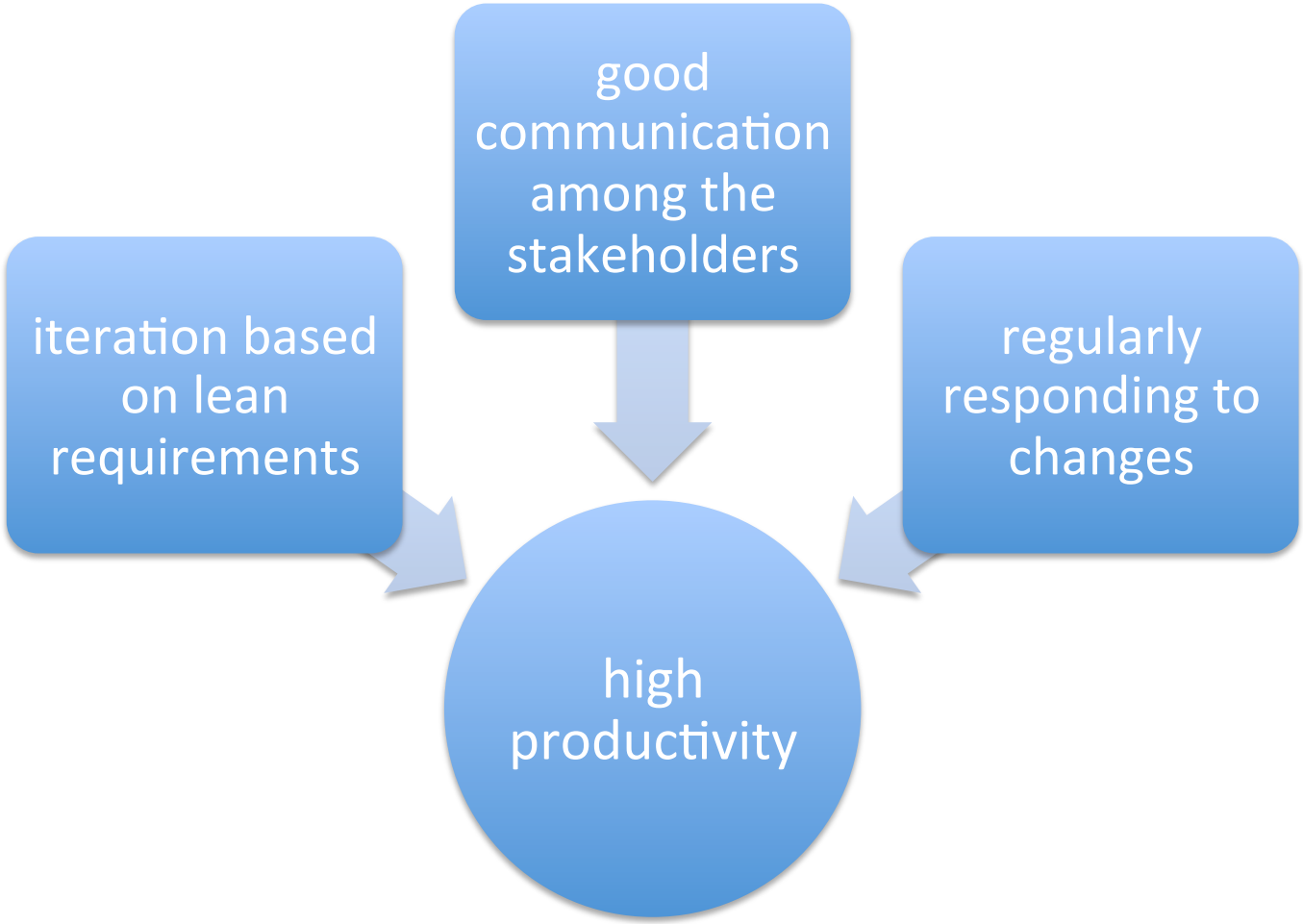
- So not just are requirements changeable, they ought to be changeable.

- Martin Fowler, “The New Methodology” -

Motivation
for Scalable Agile Formal Method



Agile Methods



Manifesto
for Agile Software Development

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

Problems of Agile Methods

Not for mission-critical software

- less reliable means of verification
- No rigorous document

Not scalable

- No explicit architectural description

Scalable Agile Method (Leffingwell, 2007)

Formal Methods

- methods to describe and verify a system based on mathematics
- originally studied in Europe in the 1970's
- many application cases to industry field
  - reinforcing safety of a mission-critical system
  - reducing regression processes in software development

Problems of Formal Methods

one new process for formal description or verification

initial training for developers

admitting the effectiveness of formal methods

hesitating to make use of them

Requirements for SAF

- Rigorous description and verification
- Test-driven development
- Object-oriented description of architecture
- Animation of specification
- Just-in-time requirements elaboration
- Automated code generation
- Internet-enabled tool for communication

Why VDM++ ?

resemblance to ordinary programming languages in description style

- familiar to most programmers

object-oriented description style

- able to describe from abstract system architecture to detailed specification

Animation of specification

- demonstration for customers to grasp the development state

Three Keys to SAF with VDM++

Automated Testing

VDMUnit: a test framework for VDM++
(transplant from JUnit developed for Java)

to be more automated

For Architectural Description (Rozanski and Woods, 2005)

- Architectural Styles
  - solutions for system-level organization
- Design Patterns
  - solutions to detailed software design problems
- Language Idioms
  - solutions to language-specific problems

Network Communication

- Agile communication
  - real-time, face-to-face, within reach
- Large Scale Developments are inevitably distributed.
- Network-enabled IDE
  - Regular IDE functions (but through the Internet)
  - Online whiteboard-like module
  - Close cooperation with video chat system

Conclusion

- Agile methods: really useful and comfortable
- But not scalable
- Agile method with VDM++: one solution for scalable agile formal development

Future Works

- More automated test framework
- Architectural style, design pattern and language idiom for architectural description
- Network communication
- All the functions are expected to be integrated in Overture (Eclipse modules for SAF)

Thank you for listening.