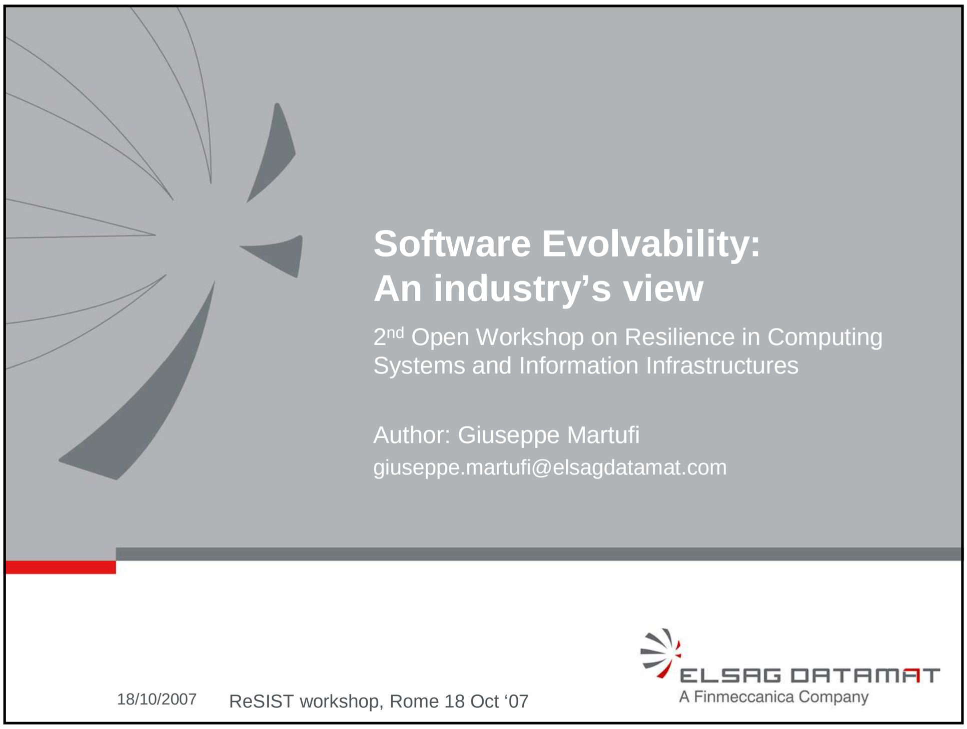
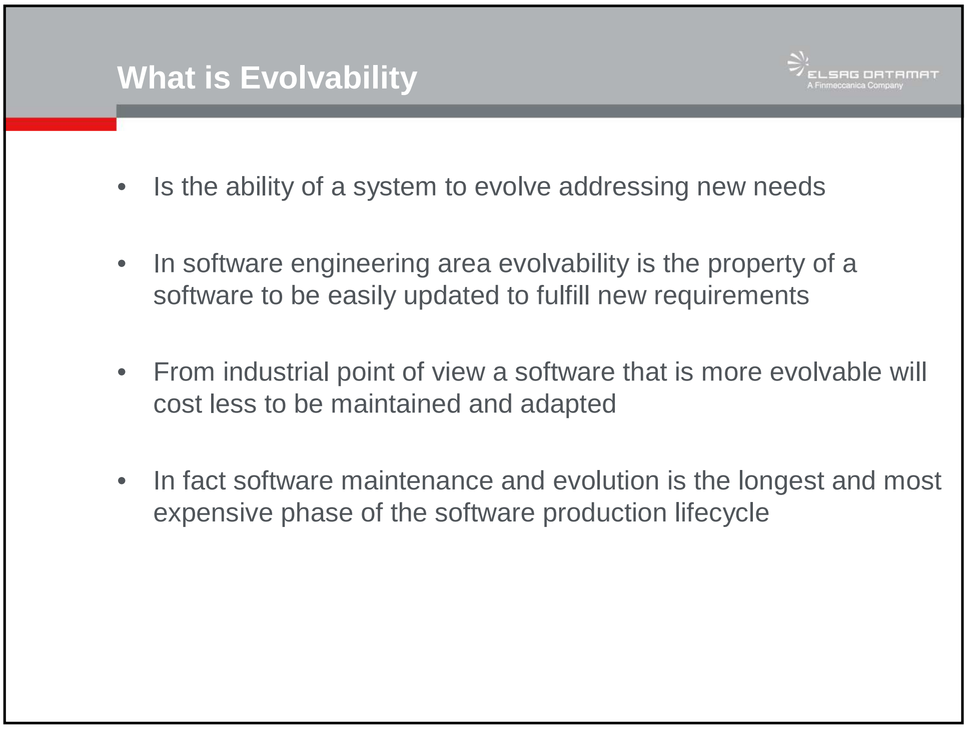


Software Evolvability: An industry's view

2nd Open Workshop on Resilience in Computing Systems and Information Infrastructures

Author: Giuseppe Martufi
giuseppe.martufi@elsagdatamat.com



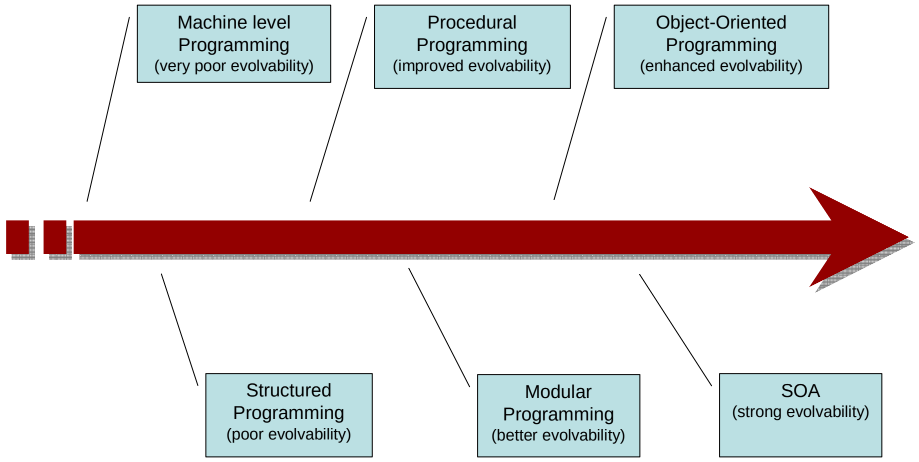
What is Evolvability

- Is the ability of a system to evolve addressing new needs
- In software engineering area evolvability is the property of a software to be easily updated to fulfill new requirements
- From industrial point of view a software that is more evolvable will cost less to be maintained and adapted
- In fact software maintenance and evolution is the longest and most expensive phase of the software production lifecycle

Main topics involved in Evolvability

- Programming Models & Software Architectures:
  - Programming Models (modularity, OO)
  - Distributed Components Architecture (RMI, CORBA, DDS, Web-Services, SOA)
- Software Engineering:
  - Development model
  - Design patterns
  - Modeling Languages (UML, SDL)
- Programming Languages (C++, Java, C#)

Programming Models & Evolvability


Machine level Programming (very poor evolvability)
Procedural Programming (improved evolvability)
Object-Oriented Programming (enhanced evolvability)
Structured Programming (poor evolvability)
Modular Programming (better evolvability)
SOA (strong evolvability)

Component based architectures & Evolvability

- A component-based application is evolvable if it is easily possible to exchange individual components without changing the others.
- Component "distance" is increasing:
  - a first stage all components were contained inside a file
  - in a second stage components have been spread out over a file system
  - the third stage is based upon components distributed over the network
  - in a fourth stage web-based service components are located in different administrated networks and domains, or the Internet (Web 2.0)

http://www.omg.org/

http://www.oasis-open.org/

New development models and Evolvability: Open Source

- Open Source is a community model
- Software development is distributed among programmers that enrich a common product
- Each programmer reuses existing code and improve components/applications based on his own needs
- Frequent sw releases and nightly builds contribute to fast evolution of a product
- Example: GNU/Linux, Apache web server, tomcat, JBoss AS

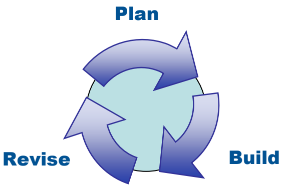
The Apache Software Foundation
http://www.apache.org/

http://www.gnu.org/

http://www.opensource.org/

New development models and Evolvability: Agile programming

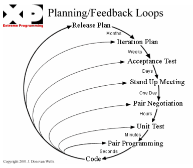
- develop software in short amounts of time (iteration)
- iteration includes all the steps of a software project (planning, requirements analysis, design, coding, testing, and documentation)
- a single iteration could not generate a product having all requested functionality, but an intermediate release
- at each iteration software product can be adapted to the emergent state of the project



http://www.agilealliance.org/

New development models and Evolvability: Extreme Programming (XP)

- XP encourages starting with the simplest solution. Extra functionality can then be added later.
- It focuses on designing and coding for the needs of today instead of those of tomorrow
- XP can produce evolvable sw:
  - a system made for today does not mean a system closed to the future
  - possible future requirements might change before they become relevant
  - an evolvable approach does not require to address today all future requirements, but to be easy adaptable to new requirements arising tomorrow



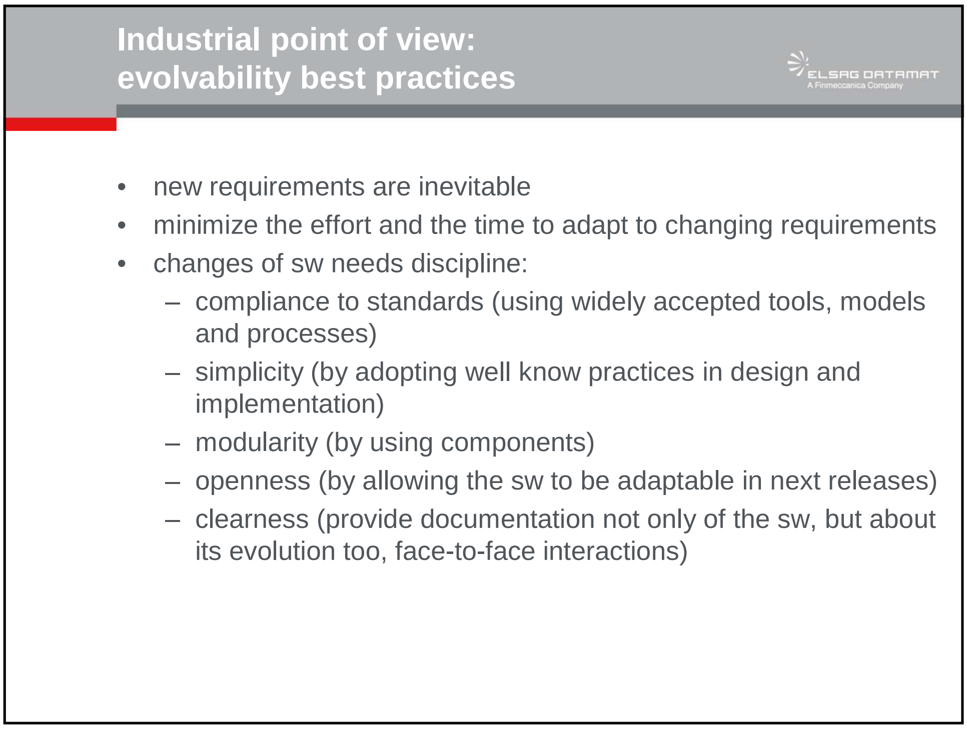
http://www.extremeprogramming.org/

Impact of sw Evolvability in Resilience systems

- an evolvable software can be:
  - easily adapted to new security requirements
  - fast to react to new threat
  - clustered and virtualized
- open sources evolution leverage to the experiences of all communities and users
- fast-iteration model reduce the time-to-react of a sw solution
- distributed component architecture spread services on the network increasing separation and reorganization

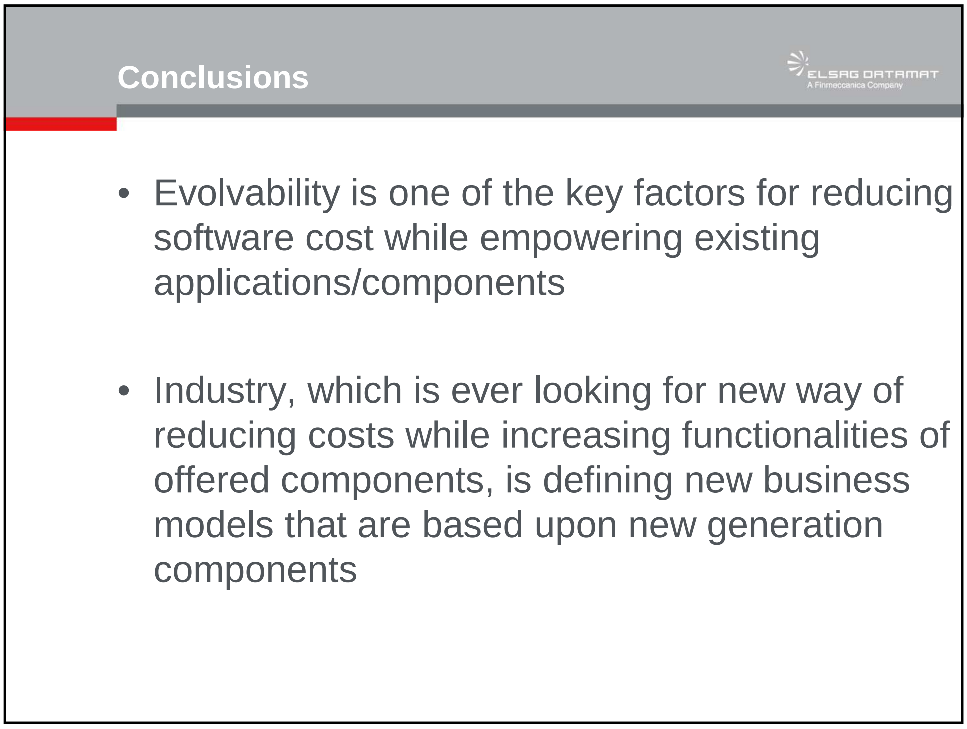
Industrial point of view

- Produce evolvable (adaptable) software allow to:
  - reduce maintenance and adaptation costs
  - improve the time-to-market
  - easy introduce changes according to requirements
- To produce evolvable products
  - modularity and component based approach are mandatory
  - adopt standard approach, models, architecture and well know design patterns
  - optimize documentation
- It does not exist the best formula for software engineering, the better choice is the one supported by experience and needs



Industrial point of view: evolvability best practices

- new requirements are inevitable
- minimize the effort and the time to adapt to changing requirements
- changes of sw needs discipline:
  - compliance to standards (using widely accepted tools, models and processes)
  - simplicity (by adopting well know practices in design and implementation)
  - modularity (by using components)
  - openness (by allowing the sw to be adaptable in next releases)
  - clearness (provide documentation not only of the sw, but about its evolution too, face-to-face interactions)



Conclusions

- Evolvability is one of the key factors for reducing software cost while empowering existing applications/components

- Industry, which is ever looking for new way of reducing costs while increasing functionalities of offered components, is defining new business models that are based upon new generation components

Thanks for your attention