Full Cycle Real Time Assurance

Dr. Sumeet Malhotra

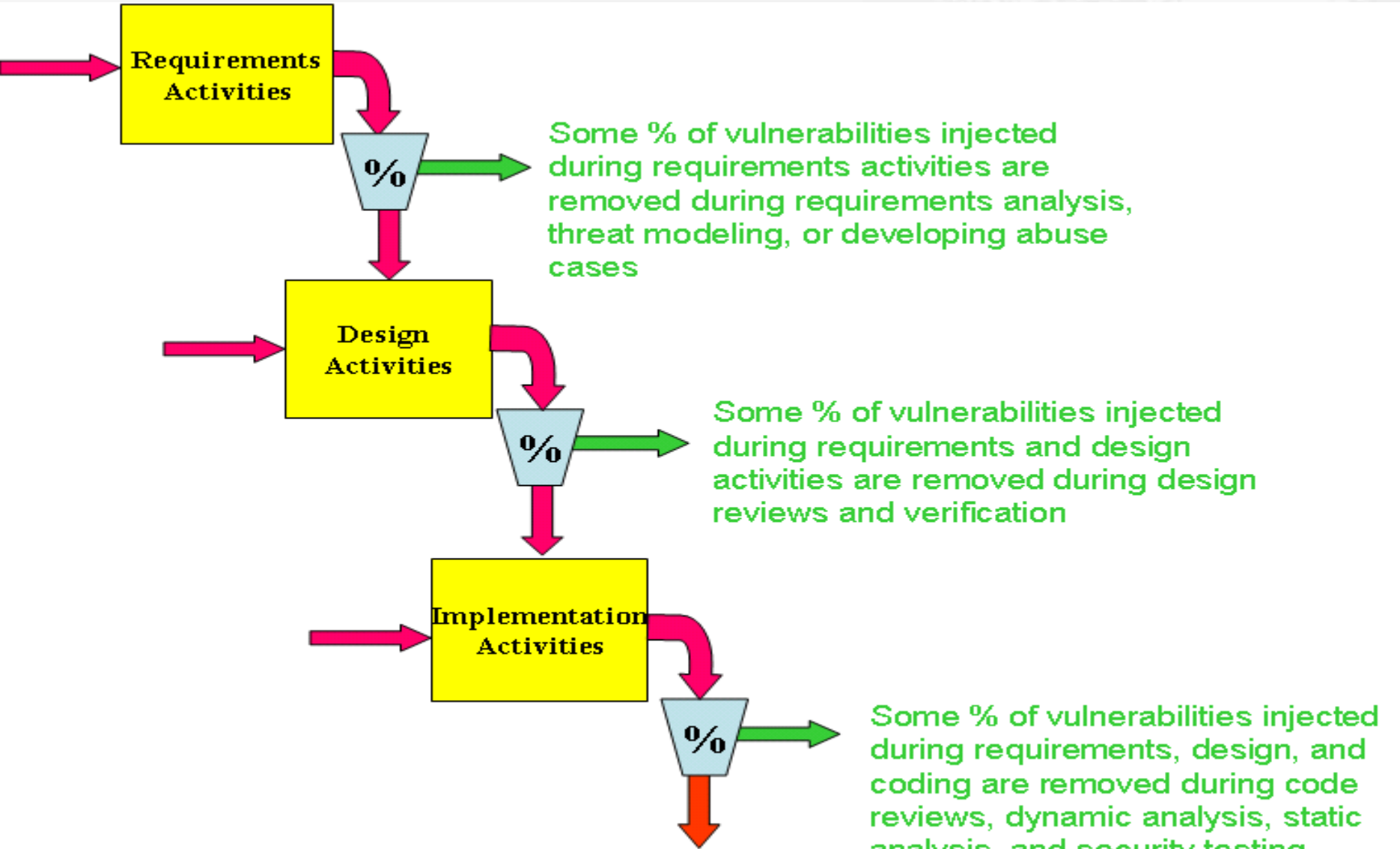
Global Director of Advanced Research & Standards Strategy, UNISYS

Member of Architecture Board, OMG

Sumeet.malhotra@unisys.com

Sumeet_malhotra@omg.org

Vulnerability Removal Filters



Correctness by Construction methodology of Praxis High Integrity Systems

- **Process for developing High Integrity Software**
- **The seven key principles of Correctness by Construction are**

1. Expect requirements to change.
2. Know why you're testing.
3. Eliminate errors before testing.
4. Write software that is easy to verify.
5. Develop incrementally.
6. Human Intelligence is critical – don’t rely on automation alone.
7. The executable software is only part of the picture. It is of no use without user manuals, business processes, design documentation, well-commented source code, and test cases. These should be produced as an intrinsic part of the development, not added at the end.

Common Criteria (NIAP Labs)

- **Section 1 = History, concepts and principles of Security Evaluation**
- **Section 2 consists of Templates for creating Protection Profiles (PP) and a Security Target (ST) document**
  - The Protection Profiles and the Security Target allow the following process for evaluation:
    - An organization that wants to acquire or develop a particular type of security product defines their security needs using a Protection Profile. The organization then has the PP evaluated, and publishes it.
    - A product developer takes this Protection Profile, writes a Security Target that is compliance with the PP, and has this Security Target evaluated.
    - The product developer then builds a TOE (or uses an existing one) and has this evaluated against the Security Target.

Common Criteria (cont)

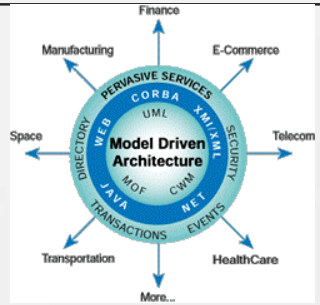
- **Section 3 includes various methods of assuring that a product is secure**
- **The included 7 pre-defined sets of assurance requirements called the Evaluation Assurance Levels (EALs) are:**
  - Evaluation assurance level 1 (EAL1) - functionally tested
  - Evaluation assurance level 2 (EAL2) – structurally tested
  - Evaluation assurance level 3 (EAL3) - methodically tested and checked
  - Evaluation assurance level 4 (EAL4) - methodically designed, tested, and reviewed
  - Evaluation assurance level 5 (EAL5) – semi-formally designed and tested
  - Evaluation assurance level 6 (EAL6) – semi-formally verified design and tested
  - **Evaluation assurance level 7 (EAL7) - formally verified design and tested**

http://niap.nist.gov/cc-scheme/vpl/vpl_type.html.

Artifacts used for Security Assurance within Agile Methods

- **Requirements**
  - Guidelines
  - Specification Analysis
  - Reviews
- **Design and Analysis**
  - Application of specific architectural approaches
  - Use of secure design principles
  - Formal validation
  - Informal validation
  - Internal review
  - External review
- **Implementation**
  - Informal requirements traceability
  - Requirements testing
  - Informal validation
  - Formal validation
  - Security testing
  - Vulnerability and penetration testing
  - Test depth analysis
  - Security static analysis
  - High-level programming languages and tools
  - Adherence to implementation standards
  - Use of version control and change tracking
  - Change authorization
  - Integration procedures
  - Use of product generation tools
  - Internal review
  - External review
  - Security evaluation

An Overview of OMG Model Driven Architecture



An Architectural Style that recommends the use of Industry Standard

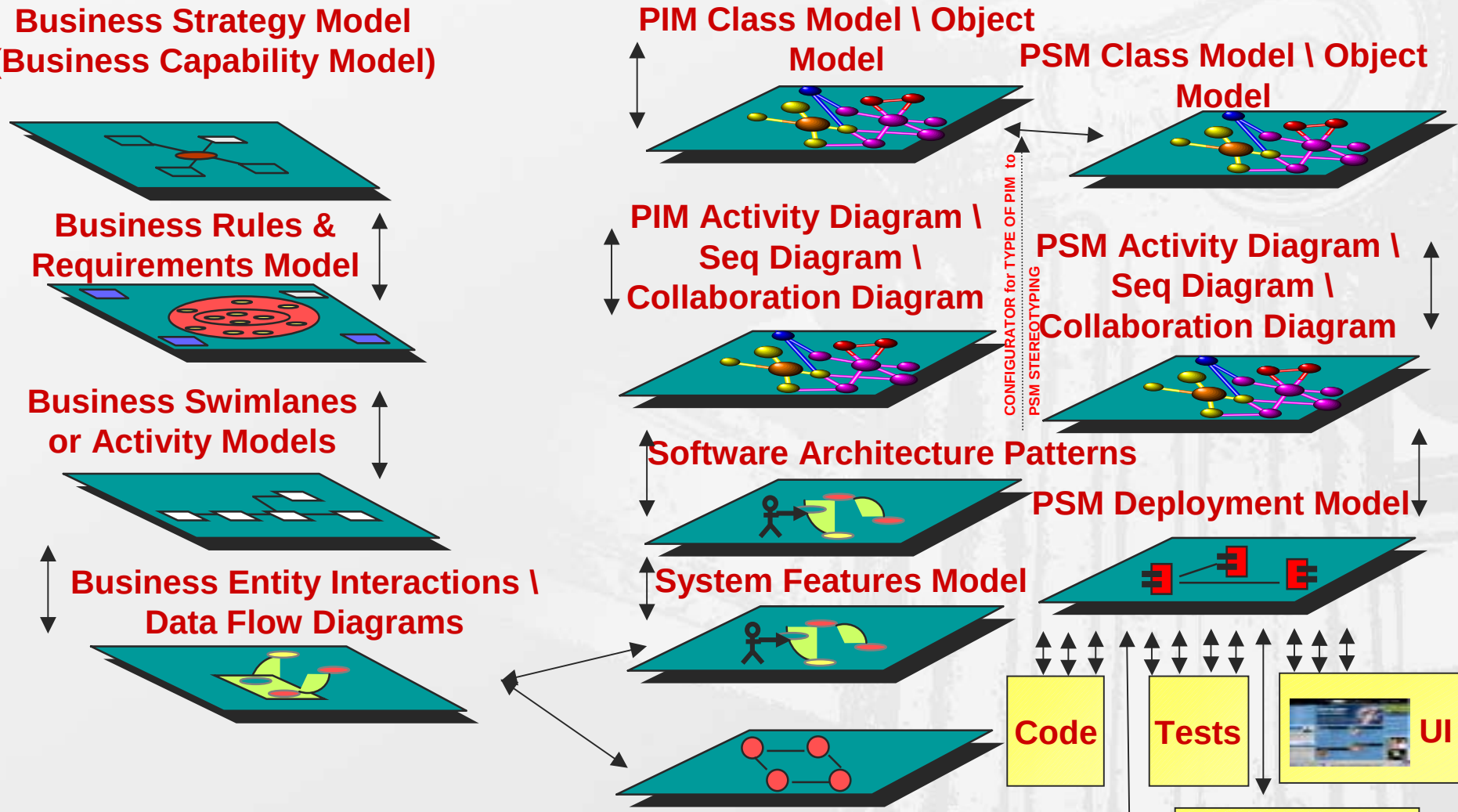
- Models (from different domains and perspectives),
- Metadata (in a standards based repository),
- Mappings (Patterns & Transformations),
- the relevant Middleware and
- Methodologies

for allowing developers and users to productively *design, build, integrate and manage applications throughout the software development lifecycle* while separating technology & business concerns.

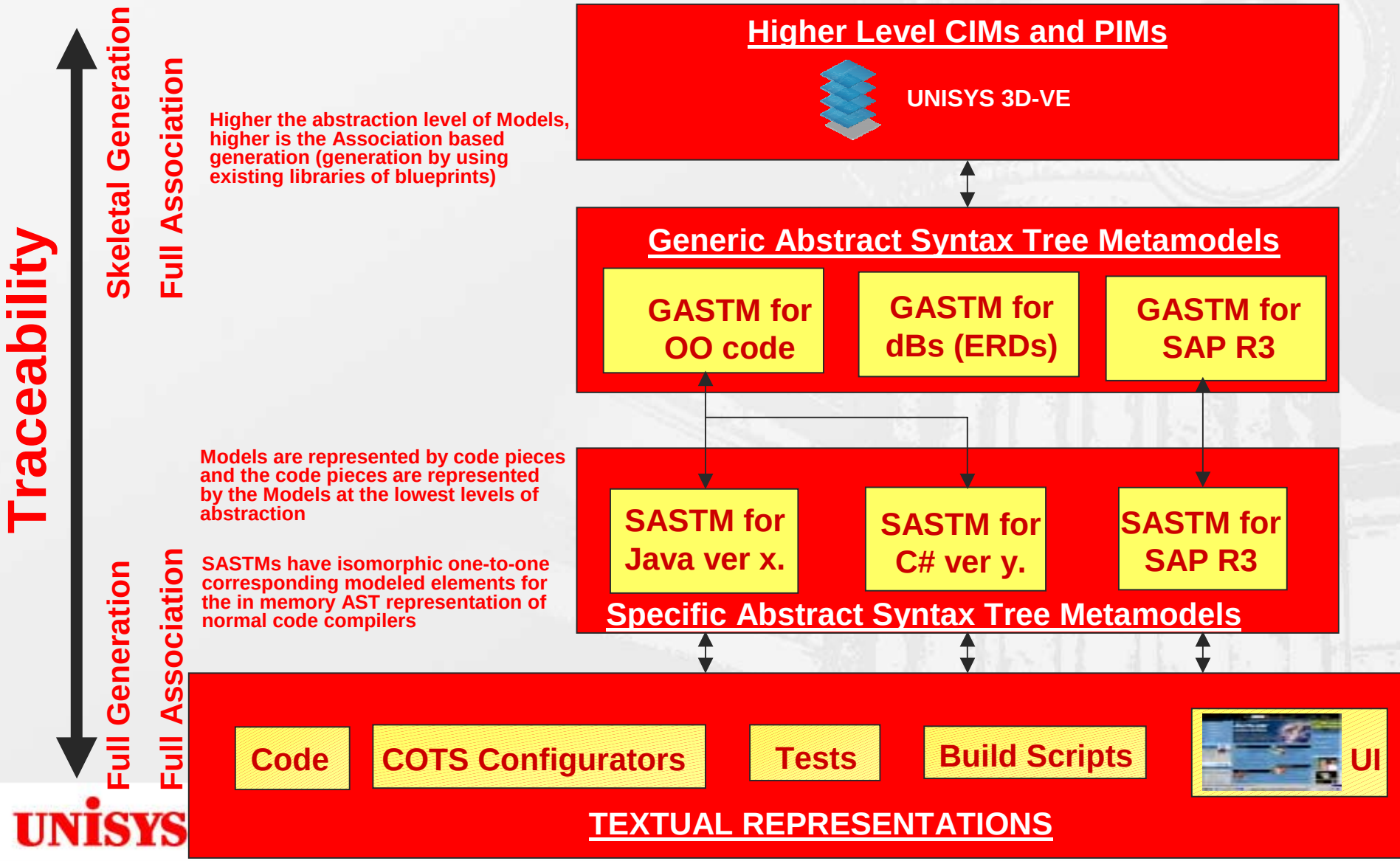
MDA

- **An eclectic integration of best practices in Modeling , Middleware, Metadata, Mapping and Software Architecture**
- **Model Driven (UML, MOF, CWM…)**
  - Platform Independent Models (PIM)
  - Platform Specific Models (PSM)
  - Mappings : PIM <==> PSM
  - Applies across the software life cycle
- **Key Benefits**
  - Ability to analyze for existence of vulnerabilities at all stages of Software Development via formal models
  - Improved Productivity for Architects, Designers, Developers and Administrators
  - Lower cost of Application Development and Management
  - Enhanced Portability and Interoperability
  - Business Models and Technologies evolve at own pace on platform(s) of choice

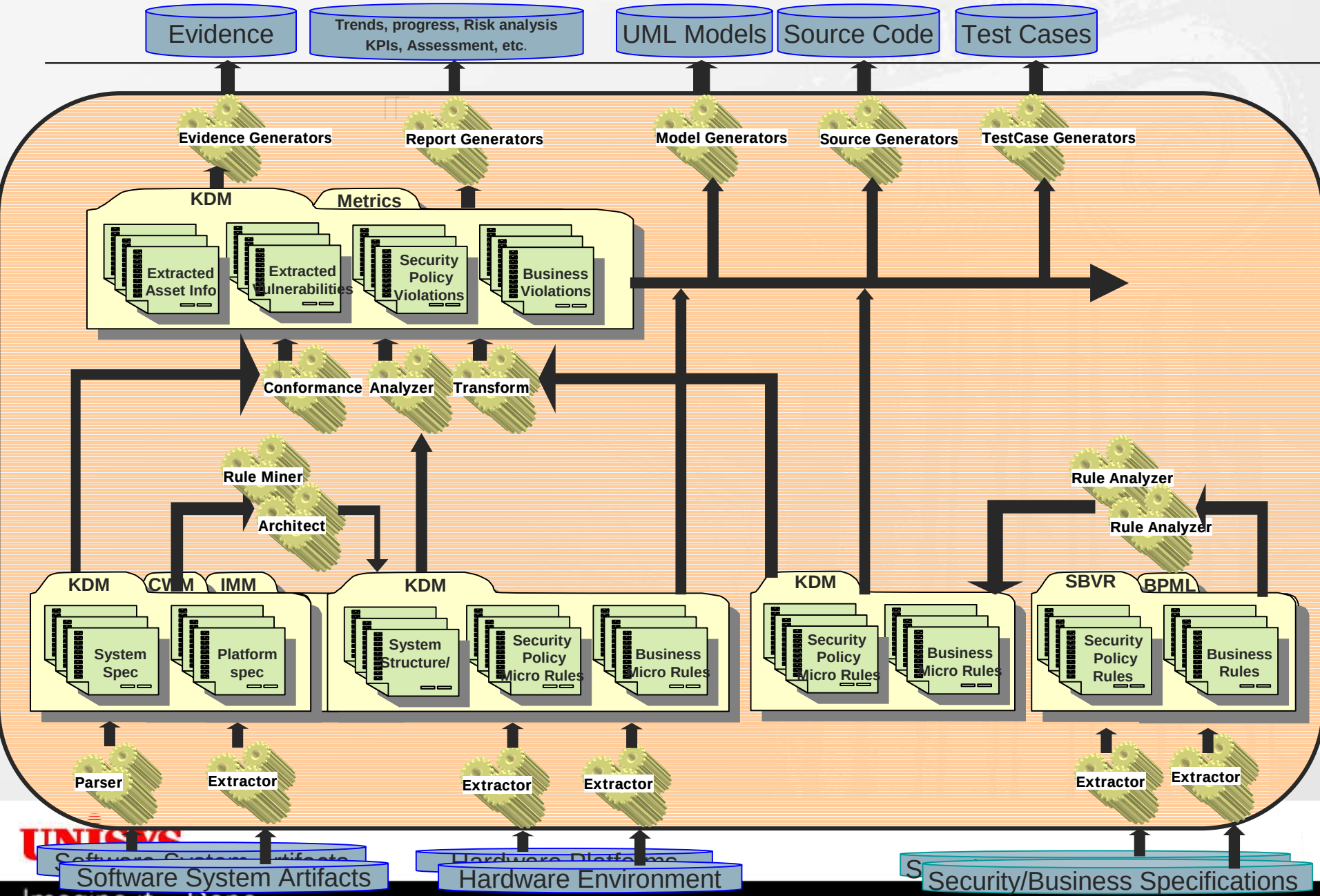
MS Software Factories & OMG MDA Guide based 3D-VE Vulnerability Analysis



Seamless Integration: Forward and Reverse Traceable Engineering



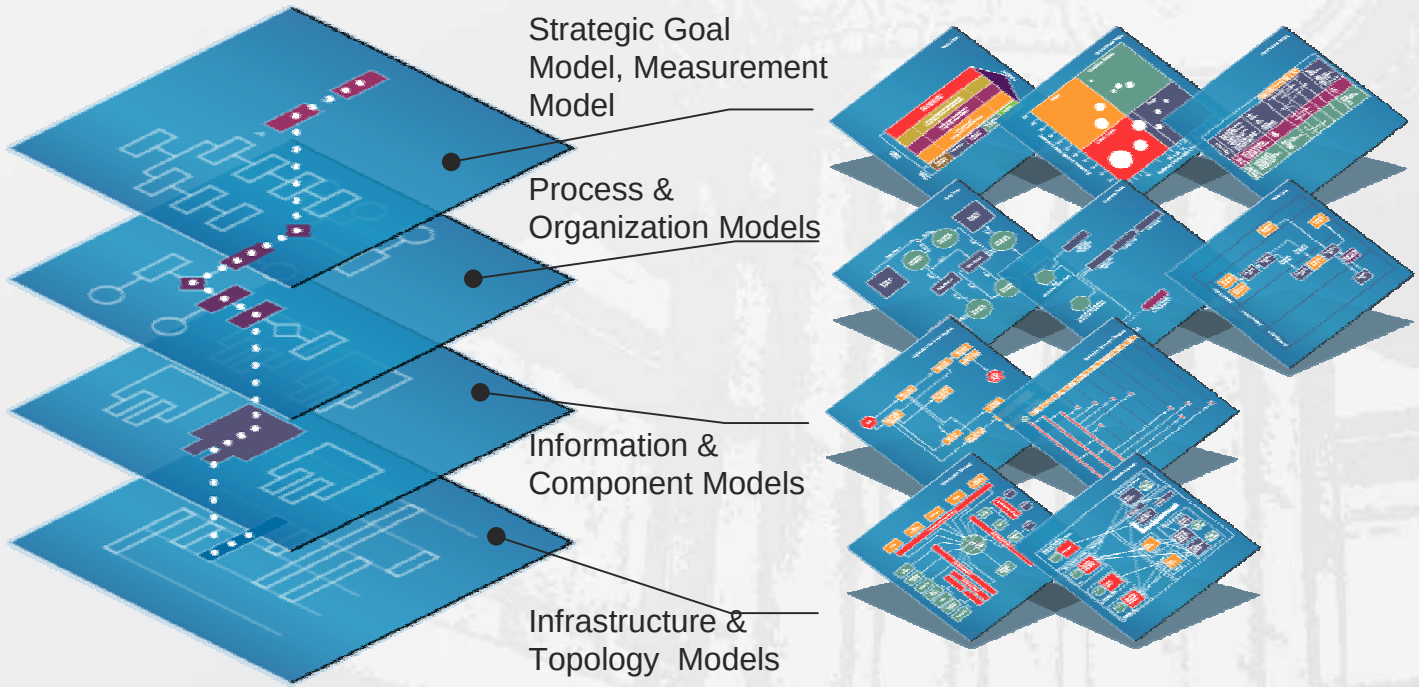
Software Assurance Ecosystem: Infrastructure with Tools



Vulnerabilities can be in anywhere - Business Models down to any Infrastructure Artifacts

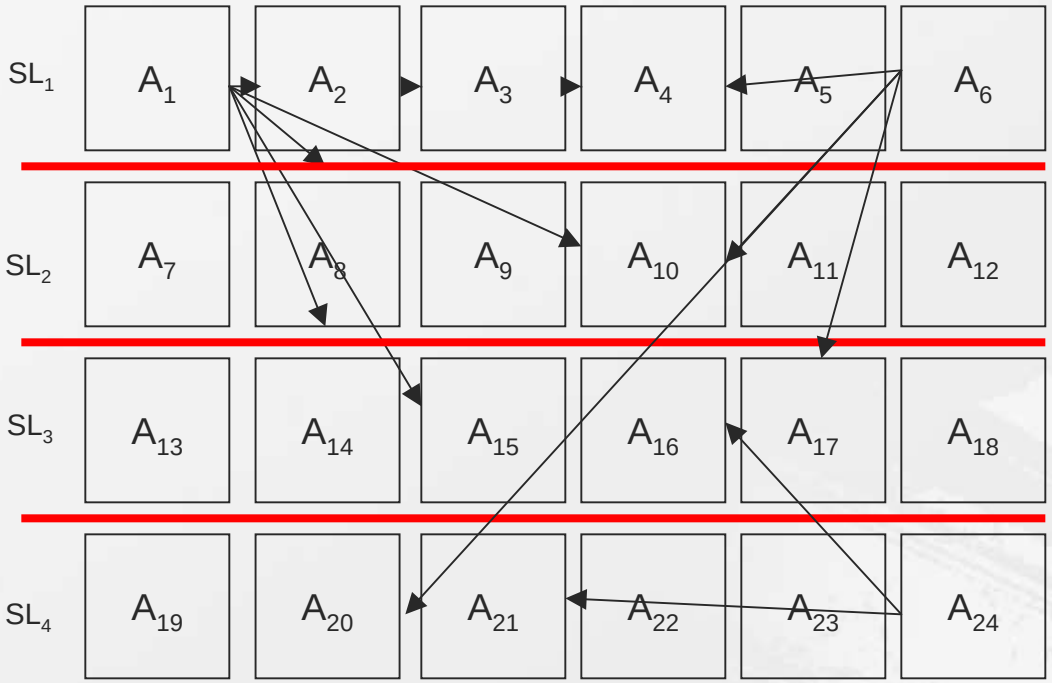
Business operations, Software and Infrastructure Operations are formally modelled in order to share and find vulnerabilities using a formal framework

- **Traceability is the backbone for maintaining and managing the analysis of vulnerabilities at any abstraction level**



If you can't model it you can't fully understand or predict behavior

Blueprints Artifact Traceability for Vulnerability Impact Assessment



Blueprint Desktop Repository

Tools
Eclipse
MDF

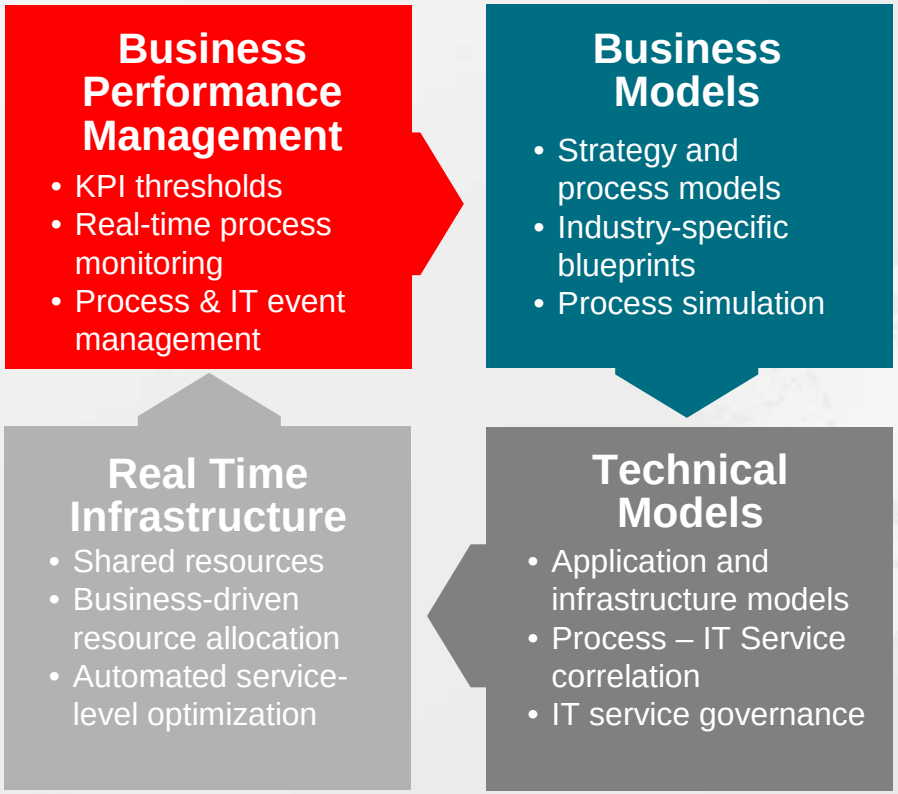
Standard
QVT
IMS

Exhaustively define at the repository level all possible traceability relationships between Blueprint Artifacts at the Repository Level

Vulnerabilities can be deduced at all levels of abstraction

Real Time Enterprise for complete Vulnerability Assessment

For Unisys, RTI is more than a set of technologies. It's integral to a framework that connects a model-based view of the business with IT and business process execution to create a closed loop that drives continuous business optimization



RTI becomes much more powerful when combined with BPM to drive the Real Time Enterprise – an organization that anticipates and capitalizes on business opportunities and threats before they happen

3D-VE Artifacts Allow Scenario, Simulation and Impact Analysis for vulnerability assessment

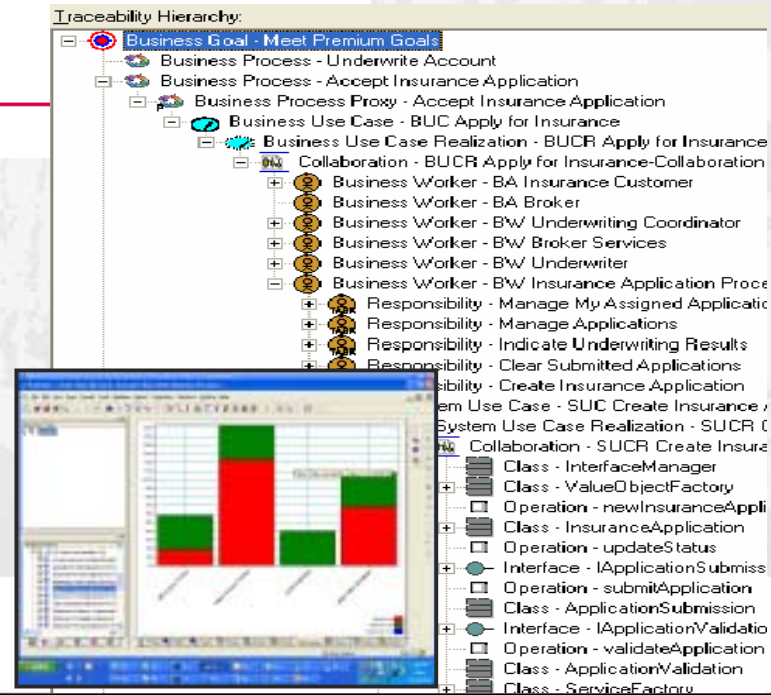
Simulation +

Layer 1	Layer 2	Layer 3	Layer 4
Business Vision and Operations Model	Business Process Model and Patterns	Functional and Application Model	Infrastructure Model
• Industry Trends Hierarchy	• Business Swimlanes	• Business Use Case Model	• Infrastructure Service Architecture Model
• Business Goals Hierarchy	• Business Events Hierarchy	• Use Case Model	• Infrastructure Cost Model
• Organizational Decomposition Model	• Business Rules Hierarchy	• Design Model	• Design View
• Industry Context Diagram	• Business Interaction Model	• Deployment Model	• Risk Analysis
	• Business Workflow Simulations	• Cost Model	

+ Integrated Costing

- ***Impact analysis and scenario planning*** **allows clients to rapidly build** ***ROI models*** **for multiple scenarios using** ***process simulation and cost estimating*** **capabilities of the 3D-VE toolsets**

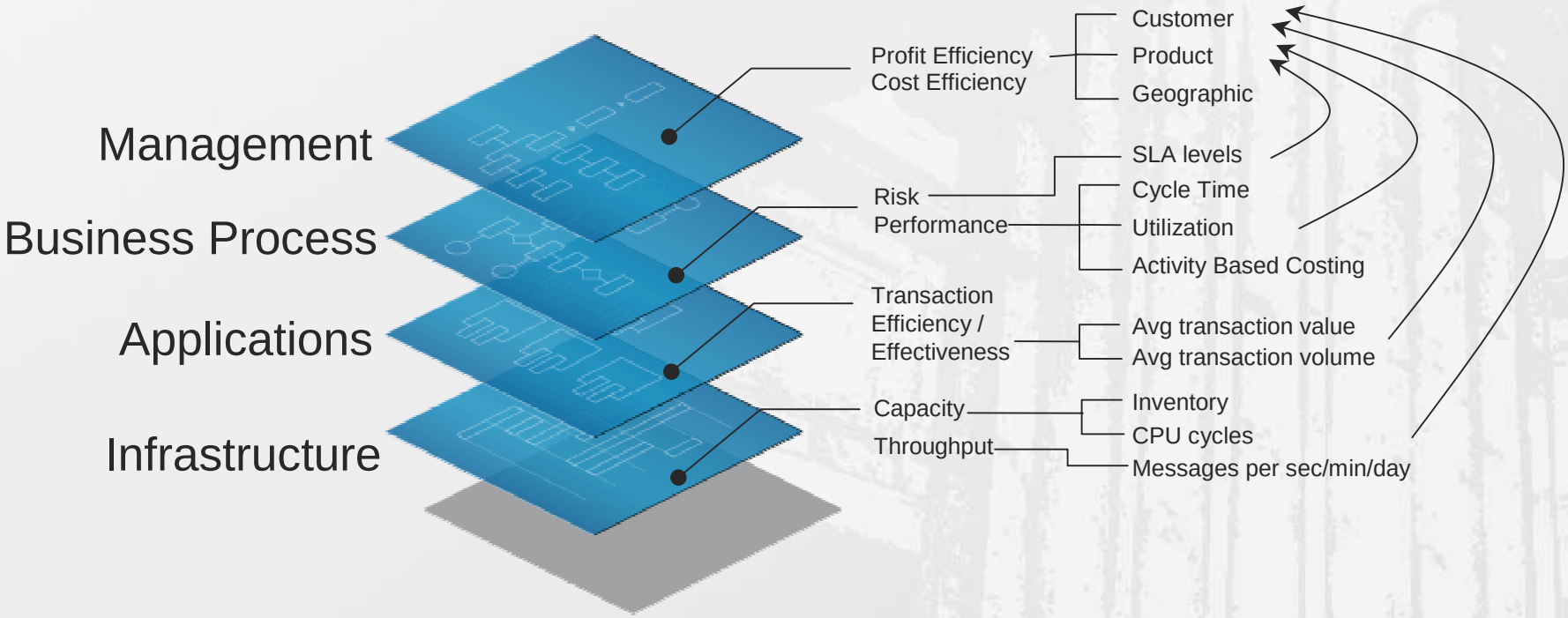
- ***Agile Change Management*** **– allows rapid changes to business process and assessment of financial impacts of change**



UNISYS

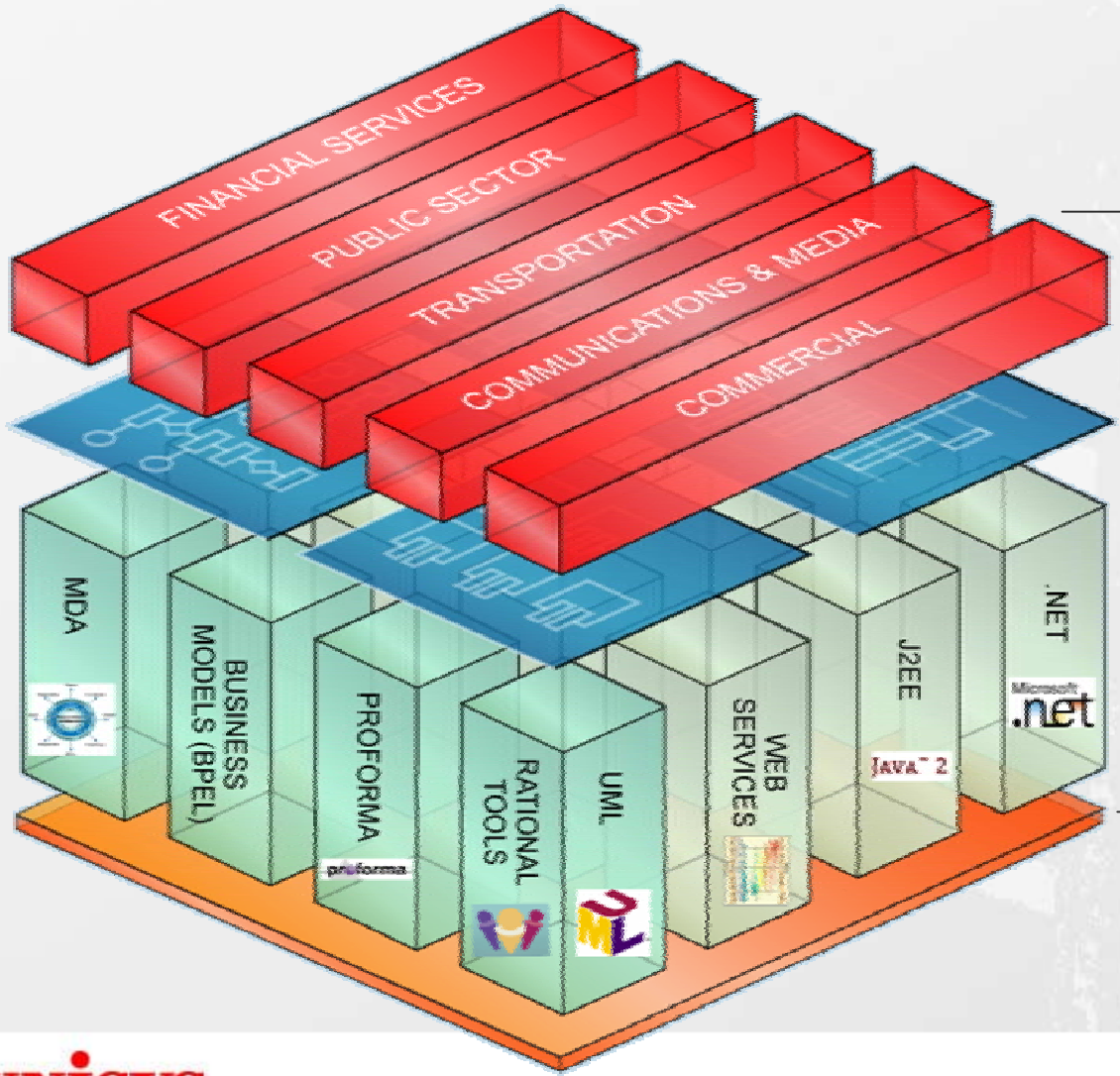
Accurate Measurement

- ***Operations should be designed to meet Key Performance Indicators (KPIs).***
- ***KPIs are derived from information in many levels of an operating enterprise***
- ***Blueprints aligns these metrics.***



You cannot improve what you cannot measure!

3D-VE Foundation For Secure Business – Vulnerability Analysis
Content at different abstraction levels



VERTICAL INDUSTRY SOLUTIONS

UNISYS BUSINESS BLUEPRINTS

TECHNOLOGY ELEMENTS

TECHNOLOGY ROADMAP FOUNDATION

References

- ISO/IEC 15288 for System Life Cycle Processes, available from http://www.iso.org
- ISO/IEC 12207 for Software Life Cycle Processes, available from http://www.iso.org
- ISO/IEC 15026 for System and Software Integrity Levels, available from http://www.iso.org
- Cleanroom Software Engineering [Linger 94, Mills 87]
- US-CERT BUILD SECURITY IN: https://buildsecurityin.us-cert.gov/daisy/bsi/articles/knowledge/sdlc/326.html

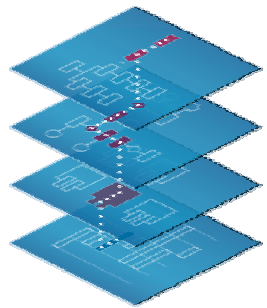
References

- Agile Alliance. *Manifesto for Agile Software Development* (2005).
- Beznosov, Konstantin. "Extreme Security Engineering: On Employing XP Practices to Achieve 'Good Enough Security' without Defining It." First ACM Workshop on Business Driven Security Engineering (BizSec). Fairfax, VA, Oct. 31, 2003.
- Beznosov, Konstantin & Kruchten,, Phillipe. "Towards Agile Security Assurance," 47–54. *Proceedings of the 2004 Workshop on New Security Paradigms*. White Point Beach Resort, Nova Scotia, Canada, September 20-23, 2004. New York, NY: Association for Computing Machinery, 2005.
- The Common Criteria Portal (2005).
- University of Wisconsin Madison. Fuzz Testing of Application Reliability (2006).
- Goldenson, D. & Gibson, D. *Demonstrating the Impact and Benefits of CMMI: An Update and Preliminary Results* (CMU/SEI-2003-SR-009, ADA418491). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2003.
- Hall, Anthony & Chapman, Roderick. "Correctness by Construction: Developing a Commercial Secure System." *IEEE Software 19*, 1 (Jan./Feb. 2002): 18-25.
- Herbsleb, J.; Carleton, A.; Rozum, J.; Siegel, J.; & Zubrow, D. *Benefits of CMM-Based Software Process Improvement: Initial Results* (CMU/SEI-94-TR-013, ADA283848). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1994.
- IEEE Standards Coordinating Committee. *IEEE Standard Glossary of Software Engineering Terminology* (IEEE Std 610.12-1990). Los Alamitos, CA: IEEE Computer Society, 1990 (ISBN 0738103918).
- Kitson, David H. "A Tailoring of the CMM for the Trusted Software Domain." *Proceedings of the Seventh Annual Software Technology Conference*. Salt Lake City, Utah, April 9-14, 1995.
- Linger, R. C. "Cleanroom Process Model." *IEEE Software 11*, 2 (March 1994): 50-58.
- Lipner, Steve & Howard, Michael. *The Trustworthy Computing Security Development Lifecycle* (2005).
- Michael, C. C. & Radosevich, Will. *Black Box Security Testing Tools* (2005).
- Microsoft. Microsoft Security Advisories (2006).
- Mills, H.; Dyer, M.; & Linger, R. C. "Cleanroom Software Engineering." *IEEE Software 4*, 5 (September 1987): 19-25.
- The MITRE Corporation. *Common Vulnerabilities and Exposures*.[NASA]NASA Software Assurance Technology Center. *Software Assurance Guidebook, NASA-GB-A201*.
- Paulk, M.; Curtis, B.; Chrissis, M.; & Weber, C. Capability Maturity Model for Software (Version 1.1) (CMU/SEI-93-TR-024, ADA263403). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1993.
- Poppendieck, M. & Morsicato, R. "Using XP for Safety-Critical Software." *Cutter IT Journal 15*, 9 (2002): 12-16.
- Redwine, S. T. & Davis, N., eds. "Processes to Produce Secure Software." *Improving Security Across the Software Development Lifecycle* (National Cybersecurity Partnership Taskforce Report), Appendix B. http://www.cyberpartnership.org/init-soft.html (2004).
- Ross, Philip E. "The Exterminators: A Small British Firm Shows That Software Bugs Aren't Inevitable." *IEEE Spectrum 42*, 9 (September 2005): 36-41.
- The SANS Institute. *The Twenty Most Critical Internet Security Vulnerabilities (Updated) – The Experts Consensus* (2005).
- Software Engineering Institute. *Maturity Profile*. http://www.sei.cmu.edu/appraisal-program/profile/profile.html (2005).
- Unites States Computer Emergency Readiness Team. *Technical Cyber Security Alerts* (2005).
- Wäyrynen, J.; Bodén, M.; & Boström, G. "Security Engineering and eXtreme Programming: an Impossible Marriage?" *Extreme Programming and Agile Methods - XP/Agile Universe 2004: 4th Conference on Extreme Programming and Agile Methods*. Calgary, Canada, August 15-18, 2004. Berlin, Germany: Springer-Verlag, 2004 (ISBN 3-540-22839-X).

What We Do.

- Consulting.
- Systems Integration.
- Outsourcing.
- Infrastructure.
- Server Technology.

How We Do It.



3D Blueprinting

What We Deliver.