About Software Assurance Ecosystems

KDM Analytics

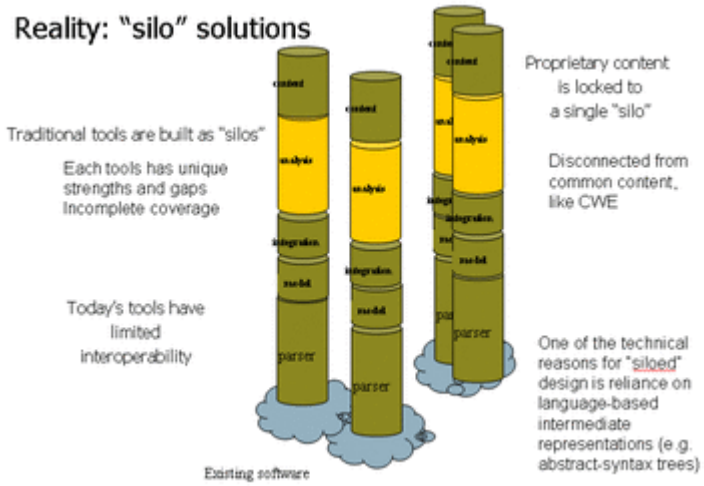
Learn more about software assurance and source code analysis tools:

- Build Security In site
- Wikipedia article on Software Assurance
- The Anatomy of a ''software analytics'' tool.

There exist multiple analytics tools for source code analysis and binary analysis. Each tool that helps understand and evolve existing software has unique strengths, but also unique gaps. As mentioned in the recent comparison of static analysis tools performed at NSA: *''Organizations trying to automate the process of testing software for vulnerabilities have no choice but to deploy a multitude of tools''.*

The following factors determine the need for deep semantic integration of source code analysis tools in the fields of software assurance, software security and software quality.

- Diversity of implementation languages
- Diversity of runtime platforms for which applications are developed
- Analysis of applications that use application frameworks, COTS components, services
- The need to add the architecture perspective to analysis of the existing software system
- The need to combine static analysis with architecture analysis and metrics
- The need to integrate static and dynamic analysis of systems
- The need to correlate software development management data from various sources



Current source code analysis tools offer little interoperability. In order to obtain a cohesive view of the security application, customers often need to do painful point-to-point integrations between these tools.

Lack of interoperability between source code analysis tools make it extremely difficult to focus community effort at developing the **common reusable content** for software assurance. Indeed, the content of each source code analysis tool is locked into a proprietary ''silo'', and each vendor is separately working on developing such content, such as the precise definitions of software weaknesses. Each source code analysis tool provides own proprietary implementations of the so-called ''checkers'', specific to the design of the proprietary analysis engine. Each ''siloed'' source code analysis tool becomes disconnected from any common reusable content created by the community effort, such as security standards, policies, protection profiles, or Common Weakness Enumeration (CWE).

The Knowledge Discovery Metamodel addresses in part the integration challenge by offering a common language-independent intermediate representation. Such intermediate representation allows logical separation of the analysis components from the parsing components and promotes a plug-and-play environment which dramatically improves the power of the individual source code analysis tools This allows significant reuse of the technologies and expertize at the analysis level. The common ontology determines the level at which tools can share knowledge about existing system, while any additional data may need to remain tool-specific.

Learn more about the design of the Knowledge Discovery Metamodel

About Software Assurance Ecosystems (cont'd)

At the semantic integration level collaborating tools need to share the same ontology related to the application. KDM provides an extensive ontology related to existing software applications. Common ontology allows developing reusable software assurance content in the form of real machine-readable artifacts that will drive source code analysis tools. Interoperability between source code analysis tools and **common reusable content** for software assurance are the main objectives of the Software Assurance Ecosystem.

The Knowledge Discovery Metamodel defines the software development database format which can be used for software asset management and software asset tracking. The KDM-based repository for a given software system can be populated by multiple software code analysis tools. KDM also allows incremental multi-phase analysis of the same software system by multiple tools, where for the advanced analysis phases the KDM repository is both the input as well as the output of the analysis. This provides better continuity and repeatability to source code analysis compared to using a proprietary internal representation of a single tool.

In order to play within the KDM ecosystem, existing tools need to implement ''bridges'' between their internal models and the Knowledge Discovery Metamodel. Compliance criteria are defined in the Knowledge Discovery Metamodel specification.

While the ''KDM Ecosystem'' is a broad community of tools, components and services built on the foundation of shared ontology, offering knowledge-based integration, the Software Assurance Ecosystem leverages the semantic integration of source code analysis tools offered by the Knowledge Discovery Metamodel and uses the common ontology to provide a common foundation for developing reusable machine-readable content for Software Assurance. In order to achieve this, the Software Assurance uses another standard from the Object Management Group (OMG), the Semantics of Business Vocabulary and Business Rules (SBVR). SBVR and KDM are designed as two parts of a unique **OMG Technology Stack** for software analytics related to existing software systems.

The Software Assurance Ecosystem brings together three separate communities :

- the formal methods community, that creates machine-readable software assurance content that can drive source code analysis tools
- the reverse engineering community, that has extensive expertise in delivering software analytics related to large, diverse enterprise software systems
- source code analysis community, that provides capabilities for automatic static analysis of software