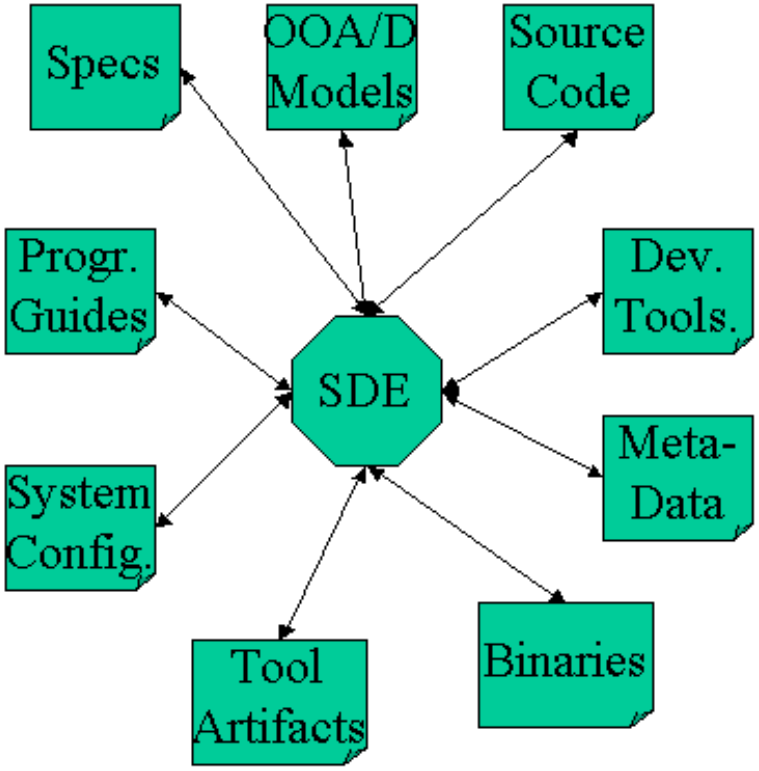
XML in Development of Distributed Systems

- Tooling
- Programming
- Runtime

Distributed System Development Environment (DSDE)

- Separation of Development, Test and Production
- Staging levels
- Integration of requirements capture, programming models and platforms and physical runtime systems

DSDE: Where it all comes together



- Schemas
- Physical formats
- Blobs
- Dependencies and Relations
- Process flows
- Context definitions

Problems in large scale system development

- Standardizing on proprietary tools causes vendor lock in.
- Proprietary formats and implicit schemas make customization impossible
- No automated setup with manually driven, closed IDEs (System Management, Security)
- Team support reduced to blob management
- manual impact control

And it's not just formats...

- To avoid hard-coding, a standard access to "informational" objects is necessary (Reference codes, properties etc.)
- EJBs (Enterprise Java Beans) are based on "separation of context": descriptive information drives runtime behavior
- Composition of components, activities and processes needs a lot of meta information
- To avoid "class explosion", TypeObject patterns can be used. They in turn require runtime context information.

Core XML features

- Web standard, forcing vendors to open their schemas and formats
- programming language independent
- separates information from behavior/code
- descriptive
- supports schema transformation
- Allows flexibility and preserves validity
- makes tools interchangeable
- human AND machine readable
- allows highly structured and linked content

DSDE tooling strategy

- Standardize on information, not tools
- All tooling imports and exports XML (MS Office 2000, Rational Rose, Object Builder, System Management, Repositories
- All meta-information is expressed in XML (Bank Information Structure, Reference Code, NLS, Logging, Object Models, Multi-entity structure,Business rule and workflow configuration, System areas, Application profiles etc.

DSDE Programming strategy

- All access to descriptive information goes through “Information Objects”
- Messages (exceptions, logging etc.) are pre-defined. Programs request empty Information Objects and fill them in.
- No static encoding of structured information in “strings”. (kills multi-entity requirement)
- Absolutely no “private” configuration files. All access goes through J2EE information servers/entity management

DSDE Programming Strategy cont'd.

- No duplication of definitions. Everything is defined in ONE place only and accessible through XML based APIs at development time and runtime.

- GUIs are built dynamically from model information, following a "stylesheet" pattern without duplication of model information in "resource files".

DSDE Runtime Strategy

- Application Infrastructure Components provide Information Object/DOM support
- Descriptive Information is served through CB services. (Availability,workload management)
- Whatever can be done manually is also available through APIs and understands XML (e.g. System Management configuration, security domain information etc.)

Where are we now?
Development Tools

- More and more tools xml/XMI enabled
- Still a lot of file based configuration, hard to maintain in production environments
- Still a lot of manual changes to configurations
- Few automatic impact or change control

Where are we now? System Management and Deployment

- Deployment of applications very cumbersome in large environments (firewalls, connectivity, change management etc.)

Can we expect help from JIRO/FMA?

Where are we now?
Meta-Data

- Few companies maintain their meta-data in proper XML formats with easy access.
- Few companies model their business processes etc.
- Few companies develop meta-data driven tools (generative programming, domain analysis)

Will the “semantic web” become reality?

Next Steps: Infrastructure

- Evaluate XML tools
- Get XML support
- Get efficient XML EJBs
- Finish the base infrastructure (AI)
- Define naming, addressing and linking elements

Next Steps: Definitions

- The information structure of large companies in XML
- Definitions for components, multi-entities, security domains, development zones etc. in XML
- Business Process definitions in XML
- Business Rule configuration in XML
- Hot Spot analysis of system and application architecture - where to use dynamic mechanisms?

Next Steps: Linking

- Create links between business processes and system management for impact analysis and control
- Create links between development tools, system management and organizational information to create isolated development zones
- Link organizational information with security and access definitions and let system management enforce the proper security modes automatically.

The XMLInformation Bus