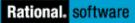
Model Management in a Team

Kim Letkeman Senior Technical Staff Member Rational Modeling Platform kletkema@ca.ibm.com

IBM Rational Software Development Conference 2008

WHERE TEAMS ARE





RU READY TO SAVE THE DAY



Agenda

Introduction

Team Development

Model Driven Development

Enterprise Model Management and Partitioning

Demo





Agenda

Introduction

Team Development

Model Driven Development

Enterprise Model Management and Partitioning

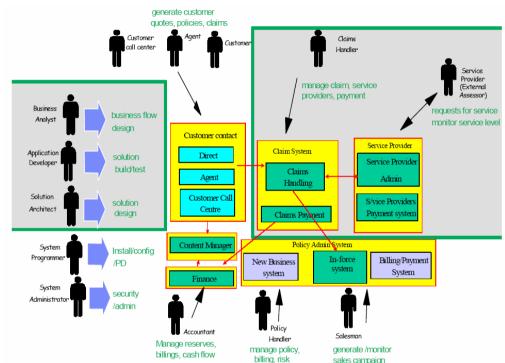
Demo





Why Model?

- Ever-increasing complexity of operational environments
 - C/C++, Java, .Net
 - Web, Handhelds, disconnected
 - Legacy integration, modernizing
- Ever-expanding choices of development solutions
 - Programming Languages, scripting Languages
 - IDE's, testing tools
- Ever-changing nature of software projects
 - Globally development teams
 - Outsourcing
 - Compliance and Regulations



More Layers More Servers More Frameworks More "Moving Parts"





Must I Model?

Modeling is the standard approach in engineering to

- Manage Complexity
- Mitigate Risk

Software development is the same as every other engineering discipline in this respect

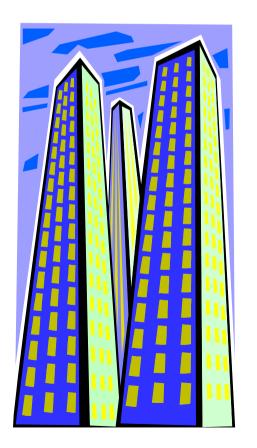
Maybe you should

Maybe you shouldn't





Maybe you <u>must</u>





Why Manage your Models?

Models are key corporate assets

• They must be stored safely and reliably.

•Every important baseline must be accessible throughout the derived product's legally mandated maintenance period.

Models are constructed in teams, with each modeler working in parallel

•*Model integrity must be maintained when parallel versions collide upon check-in.*



What is Merging?

For this discussion:

- **Merge** will mean 3-way merging to resolve parallel versions of a model when the second and all subsequent versions are checked into a repository
 - This operation is automatically invoked by all supported repositories
 - Merge uses the identity of objects to align them for comparison
 - "Compare with ..." commands within the Eclipse environment use a variation of merge
- **Fusion** will mean 2-way structural merging to compare two models and capture structural differences from a source model into a target model
 - This operation can be performed ad hoc, by invoking the combine models command
 - Or it can be automatically invoked at the end of a code to model transformation
 - Fusion uses the name of an element and its path (list of container names) to align elements for comparison



Agenda

Introduction

Team Development

Model Driven Development

Enterprise Model Management and Partitioning

Demo





What is Team Development?

Modelers generally work in parallel on a common set of artifacts

- Manual coordination impedes the smooth flow of changes, so automatic coordination for parallel development is extremely desirable
- RSA directly supports team development through ClearCase, Team Concert and CVS with fully automatic coordination of parallel development
- RSA also supports ad-hoc modeling during the transition to more formal team development practices

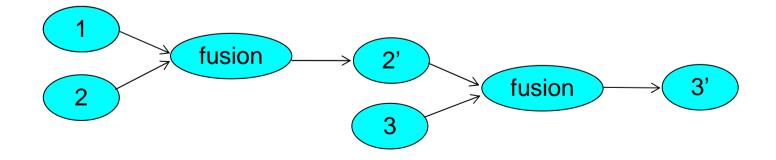


What is Ad Hoc Modeling?

During the early stages of a project, it is common for several modelers to create similar models with common objects

Chief characteristics are:

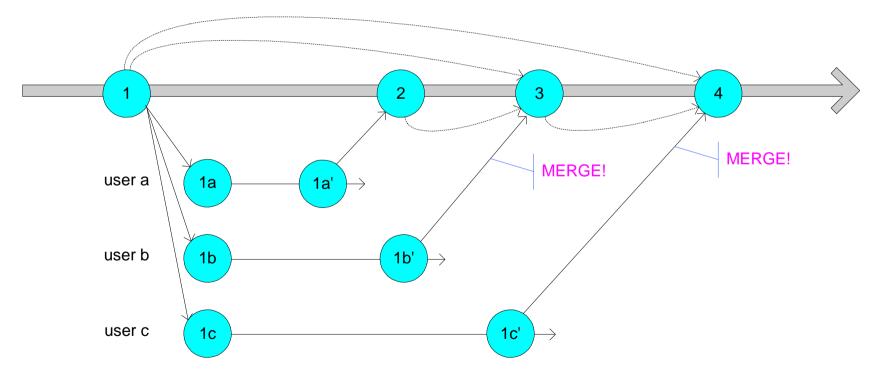
- the lack of any formal version control;
- models that were created independently;
- models that do not share ancestry (although copying models to seed other modelers is also supported)





What is Formal Version Control?

ClearCase, Team Concert and CVS automatically detect the need for a merge! Typical behavior in mature projects





Supported Team (SCM) Repositories

ClearCase

- Pessimistic locking, must checkout each artifact
- Can check out all artifacts
- Runs unconnected with automated hijacking
- When reconnected, ClearCase will checkout those files that have been hijacked

CVS

- Optimistic locking
- File changes are seen at synchronization time

Rational Team Concert

- Optimistic locking
- File changes are noted in real time or when polled
- Integrated collaboration tools
- Integrated work item management
- Integrated process and work flow management

AM11



Agenda

Introduction

Team Development

Model Driven Development

Enterprise Model Management and Partitioning

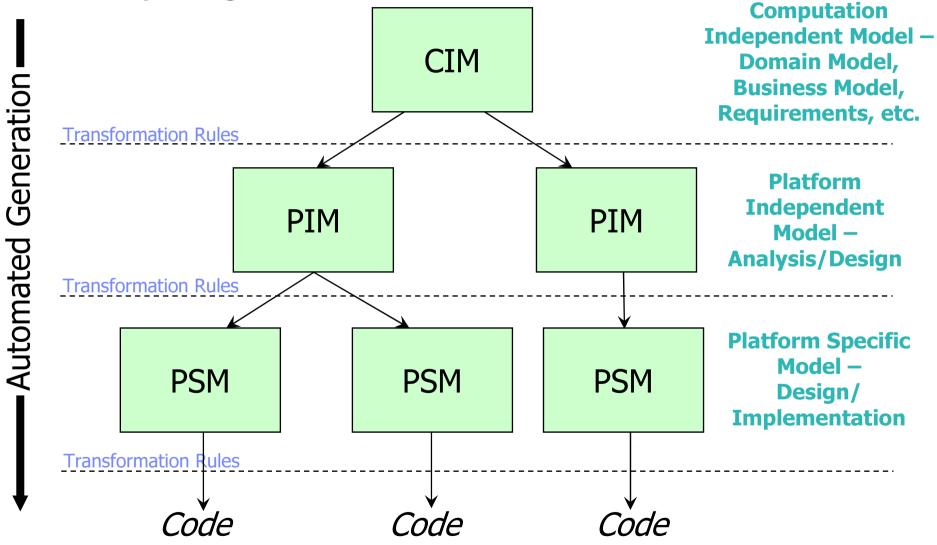
Demo



IBM Rational Software Development Conference 2008



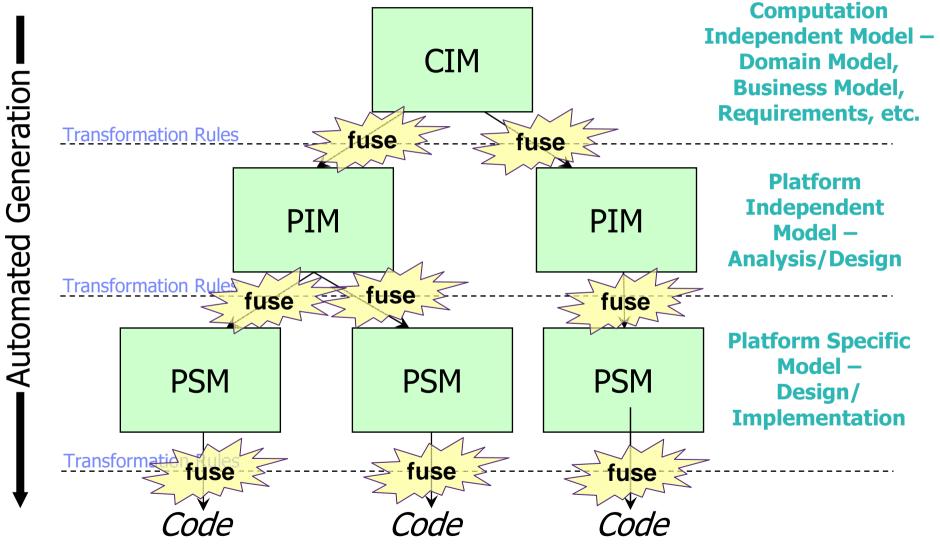
MDA Layering



AM11



MDD with Fusion



AM11



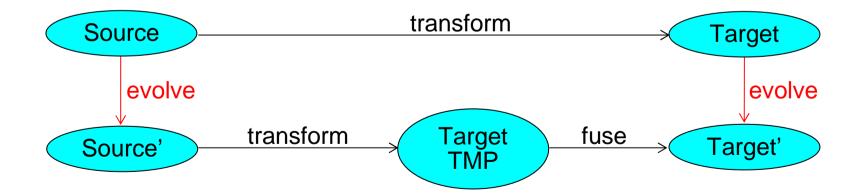
Fusion at the Transformation Target

Iteration is possible at every transformation stage

Change the source model, and you must transform again

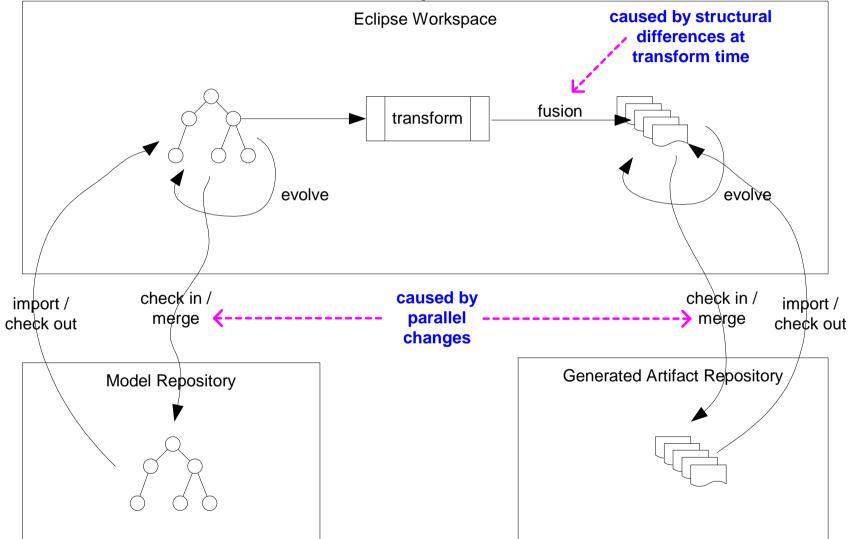
Change the target (in parallel,) and the next transformation will have to be merged

There is therefore a **fusion** step in each transformation





Model Driven Team Development Artifact Flow





Agenda

Introduction

Team Development

Model Driven Development

Model Management Best Practices

Demo



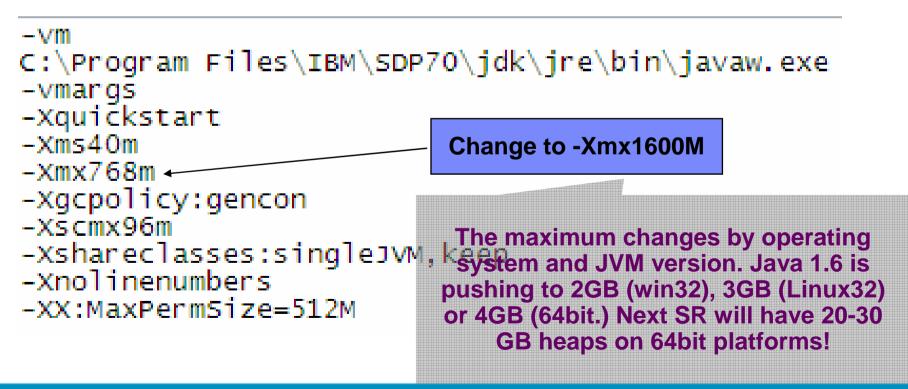


Tip: Allocate Sufficient Heap Space

The default heap space for RSA is not sufficient for the largest models

The maximum heap space of 1.6GB will help with merging of these models

Change this parameter in your eclipse.ini file (in the installation folder):





Tip: Consider Disabling the Swap File

When a workstation has 2GB or preferably 3GB or more, it is a candidate

Windows performs dynamic paging of memory to disk to simulate a much larger memory space and allow many applications to share the CPU, but paging is slow, manifesting as random pauses or slow response when bringing a new application to the foreground

Java	app	lications	aggravate	the	situation

Paging can be turned off in the System Control Panel.

Drive [Volume Label]	Paging File Size (MB
C:	
Paging file size for sel	ected drive
Drive:	C:
Space available:	62493 MB
O Custom size:	
Initial size (MB):	
Maximum size (MB):	
O System managed :	size
 No paging file 	

WARNING: Smaller memory sizes or running several concurrent applications can cause an out of memory error and applications may crash. This technique should be considered on a case by case basis for advanced users only! The author has used the technique since 2004, but it carries some risk.



Best Practice: Strictly Control Custom Profiles

Extend the UML meta-model with a custom profile to adapt a model to a domain A custom profile *changes a model's meta-model once applied*

• At this point, the profile is as important as the UML meta-model itself

Separate the custom profile's life cycle from the models to which it is applied

Follow the formal process for custom profile development as defined in the profile authoring guide and the team development guide

- http://www.ibm.com/developerworks/rational/library/05/0906_dusko/
- http://www.ibm.com/developerworks/rational/library/05/0823_Letkeman/

Deploy your custom profiles as plugins with pathmaps!

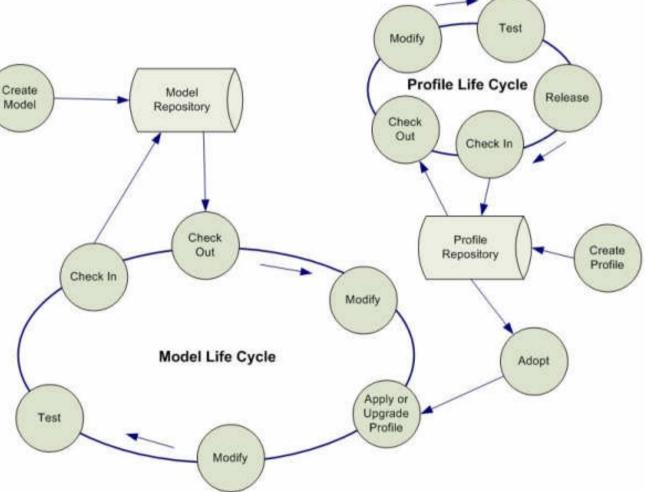
 Deploying profiles that are under active development into a workspace project alongside the models under development is an anti-pattern – it promotes loose upgrade and migration practices, which leads to subtle model corruption that shows up during merging sessions



Best Practice: Separate the Custom Profile and Model Life Cycles

It is *very important* that these life-cycles be separated.

- Profiles should be stored separately from models.
- Failure to follow these guidelines can result in models that cannot be opened.
- More information is available in Part 6 of the compare merge article series.

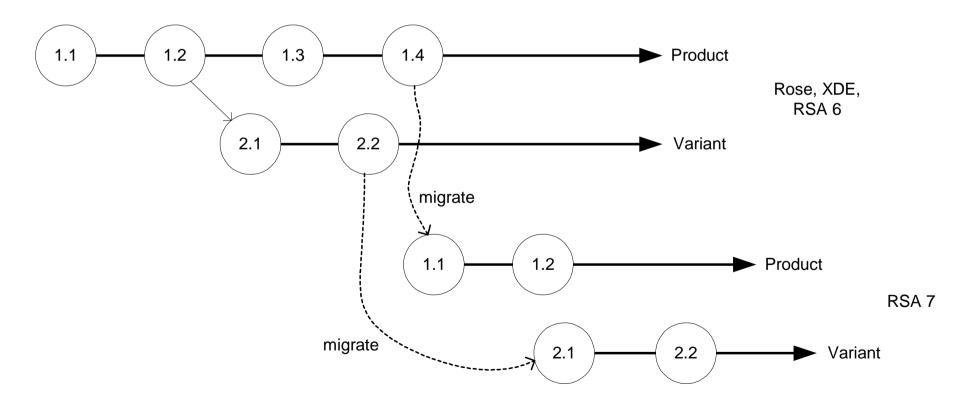




Tip: Multi-stream Model Upgrades

Variations of a set of models can exist on multiple streams or branches

When these models are upgraded to RSA 7, many thousand new elements are created more than once, and they **do not match**





Tip: Multi-stream Model Upgrades (2)

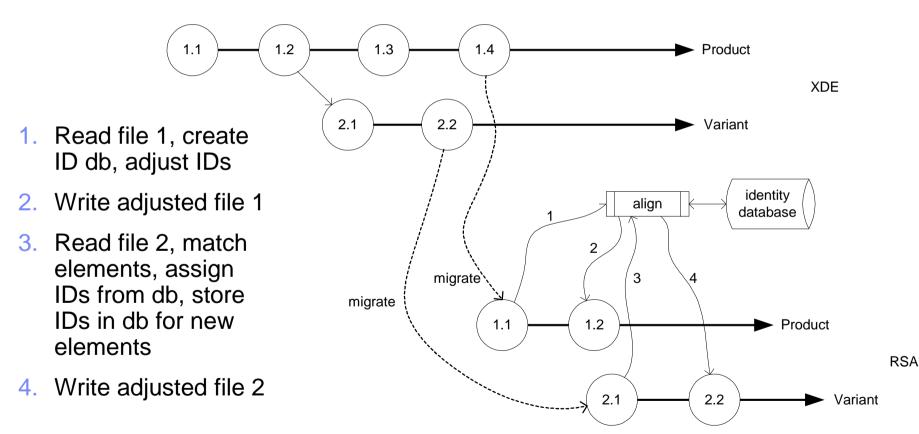
Upgrading from XDE or Rose to RSA, or from RSA 6 to RSA 7 changes UML and notational meta-models

- These meta-model changes require the creation of many new elements that did not exist in previous UML generations we've seen 10,000 per model
- These new elements have new identifiers and the same new elements are created in each stream; each have a new unique identifier *but these will not match during a compare!*
- RSA 7 includes technology to realign identities in a series of models, effectively recreating the element ancestry relationships from stream to stream ad infinitum
- After running the model alignment tool between two such streams, any two models will have the same identifiers for any element that is intended to be the same, thus superfluous differences are removed



Tip: Multi-stream Model Upgrades (3)

The model alignment tool workflow looks like:





Best Practice: Understand the Factors that Affect Physical Model Organization

Team structure

- Hierarchy within enterprise, all sharing common models?
 - One enterprise model set in a UCM environment
- Separate teams delivering components to a common system?
 - Multiple smaller model sets in a distributed SCM environment

Projected model size

- Small models?
 - Any organization will work
- One huge model?
 - Fragmented will perform best



Best Practice: Understand the Factors that Affect Physical Model Organization

Computing power available

- Resource constricted machines?
 - Fragments will perform best

Preferred operational mode

- Prefer to share a stream and operate live on one model copy?
 - Fully fragmented (exploded) model on a single shared integration stream with dynamic views and enforced reserved checkouts
- Prefer to isolate modelers from one another and merge the work later?
 - Consider UCM or any equivalent branching mechanism with fragmented models
 - Smaller models need not be fragmented



Best Practice: Strong Ownership

When people work on the same packages and diagrams, model element conflicts are inevitable

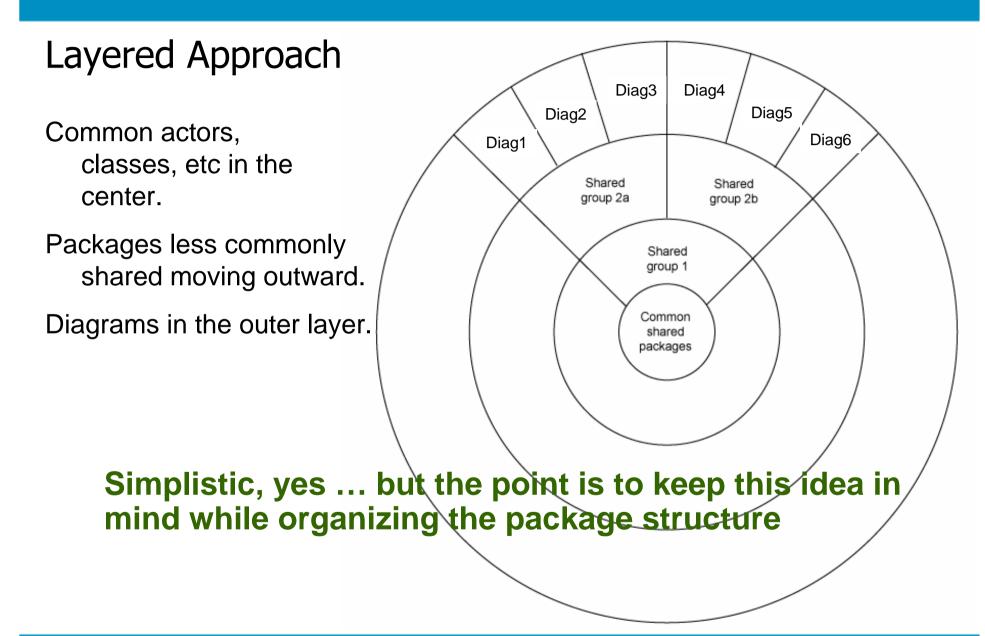
- When conflicts at the element level occur, the SCM system can no longer get a successful automatic merge; instead, a visual merge is launched and user intervention is required
- If strong ownership is practiced, and people's work is separated more effectively, most merges are automatic and convenient
- It is not necessary to fragment a model to practice strong ownership, simply organizing the package structure effectively can enable this behavior



Best Practice: Layer Your Model Packages

- Think of a target, wheel or onion. Higher levels of sharing occupy the center, while packages with lesser sharing move further outward in rings or layers.
- Ideally, references to elements always flow inward between rings, never sideways within a ring.
- Diagrams tend to occupy the outer ring in separate "spokes."
 - Packages used by diagrams tend to flow from less shared to more shared.
- When correctly partitioned, separation into models is easily accomplished if necessary. Strong ownership is easily applied as well.







Best Practice: Fragmentation of Large Models

Break the model into tiny fragments down to classifiers and diagrams

Does not preclude appropriate package layering

Supports strong ownership

Very small memory footprint

a small number of fragments in memory at any one time

Complete hierarchy is visible within the model explorer

• In 7.5, expand in place for imported packages makes this a reality for separate models as well, allowing both techniques to be mixed and matched

With strong ownership, file by file merge is safe when certain rules are followed

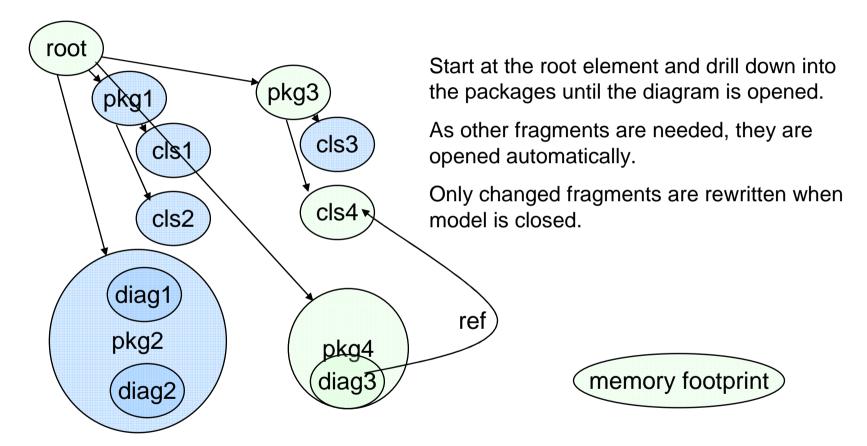
Full-context merge is perfectly safe under all circumstances

In 7.5, full-context sparse merge is available so that arbitrarily large models can be merged!



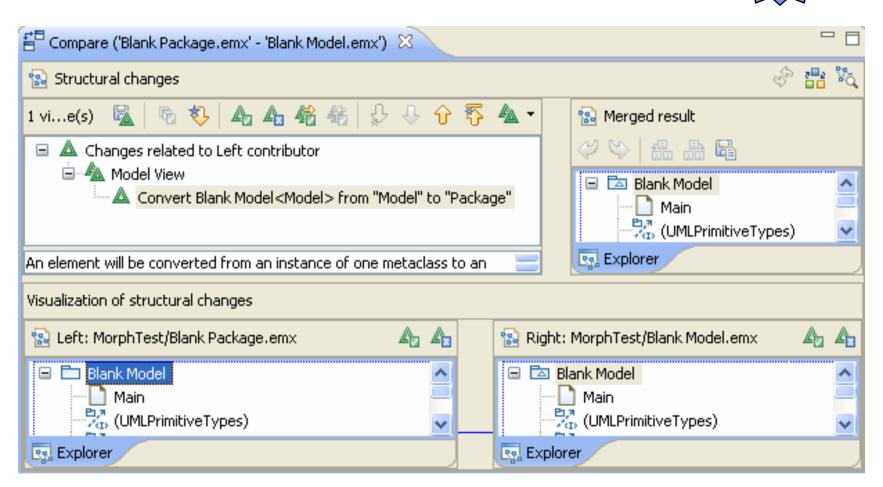


Fragmentation Memory Footprint





New in 7.5: Shared Fragments are Back!





Best Practice: Shared Fragments

It is once again possible in release 7.5 to separate your packages into independent fragments.

- Packages can, in fact, be the root of a fragmented model made of up many separate files.
- Packages can be imported into several models, effectively sharing them across models

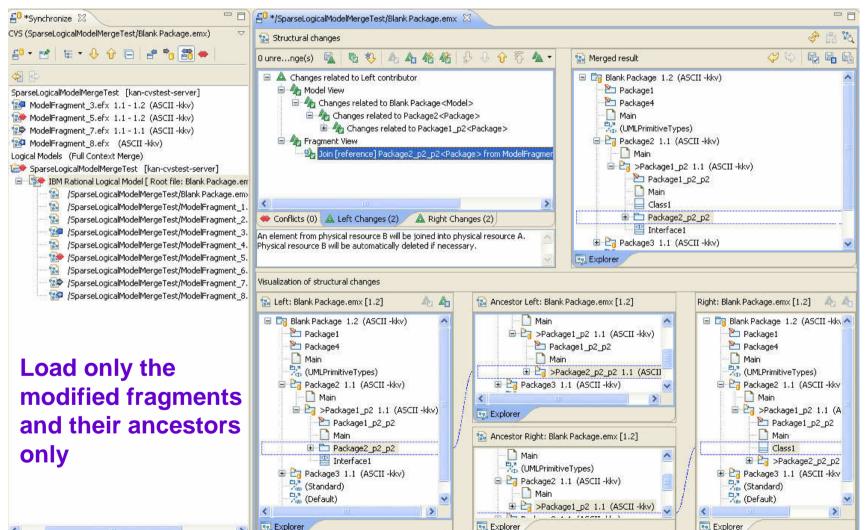
Packages will be expanded in place in the project explorer

IBM Rational Software Development Conference 2008



Sparse Logical Model Merge





AM11



Enable Sparse Logical Model Merge option

Available in RSA 7.0.0.7, RSD and RSM 7.0.5.2, and RSA 7.5

Works in findmerge and UCM with CCRC 7.1

Works in CVS

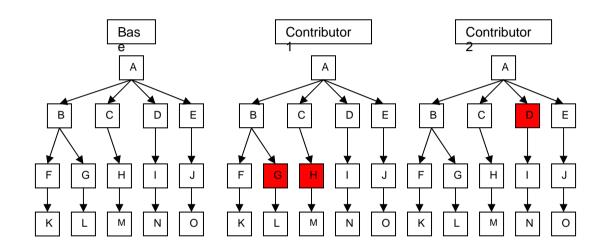
For RTC 1.0, disable logical model mode instead, and practice strong ownership!

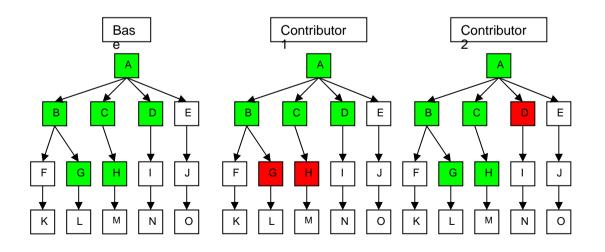
🖨 Preferences	
type filter text	Modeling Compare/Merge 🧔 🗧 🖕
 General Appearance Capabilities Compare/Patch Modeling Compare/Merge Combine Models Merge Compare/Merge Server Delta Tree Configuration EMF Compare/Merge Content Types Editors Keys Markers Network Connections Perspectives Search 	 Set focus on the default view mode when navigating the structural changes Note: The default view mode is the first tab within the pane. This preference will take effect on any open compare/merge sessions. Automatically accept resolvable differences for repository merge session Logical Model Settings Disable Logical Model Synchronization Note: Select this option to individually update, commit or synchronize emx and efx file artifacts. When using this option, it is highly recommended that fragment creation and absorption is not performed in parallel with other users else it can result in corruption of the model. It is also recommended to avoid refactoring of the model elements between fragments. When necessary, perform these changes in a locked repository.
Startup and Shutdown Web Browser Workspace Active Correlation Technology Agent Controller Analysis Ant C/C++ Data Management Corre Diagram	 Enable Sparse Logical Model Merge Note: For a large model with many fragments, full context (logical model) merging can take significant time because of the need to create three complete models as inputs, and to create a fourth complete merge model as output. Sparse model merge speeds this up significantly, as only those fragments that actually changed are loaded, along with their parent containers up to the root of the logical model. In this way, it is possible to perform a full-context merge with all the benefits of model-integrity protection but with much faster performance because only the change set is loaded for each copy of the model. There are side-effects, however, notably the appearance of broken references where fragments are missing. This does not affect the merge result.



How Does Sparse Merge Help?

By merging a complete subset of the models! All changes appear, all unchanged files are ignored.





- Full subset identified for merge.
- All the white files left out. This scales to any sized models, since the majority are ignored!

IBM Rational Software Development Conference 2008



NEV

Best Practice: Filters, Sorters and Groupers!!

No longer a slave to excessive deltas, it is now possible to suppress **noise**

Ships with many useful filters

Customizable

Extensible through extension point

Can sort by delta type or object

Can flatten delta list!

🚰 Compare ('Blank Model.emx' - 'Blank Model.emx' - 'Blan	nk Model.emx') 🔀	3	
😰 Structural changes			S 📑 🇞
1 unresolved conflict(s) and 25 visible change(s)		🔝 Merged result	4 🗟 🖪
🖾 🔁 🎝 🗛 🖧 🏍 🗳 🗘 🗘 🖓	A	🖃 🖾 Blank Model	_
 Changes related to Left contributor Delete change Delete change Delete «documentation»"attribute1 doc" Modify change Modify Blank Model<model>.name : Stri Modify Blank Model<model>.name : Stri Modify «documentation»"1\r\n2\r\n3\r\n Add change Add Artifact1 <artifact> to Blank Model</artifact> Add «documentation»"1\r\n2\r\n3\r\n4\ </model> </model> 	Filter out Move Changes Filter out Fragment Join Changes Filter out Fragment Separation Changes Filter out Diagram Position and Size Changes Filter out Diagram Changes Filter out Stereotype Application Changes Filter out Documentation changes		≡ antatior ()
Conflicts (1) Left Changes (8) Right Change	✓ No sorting Sort by model element name Sort by Delta Label		.
Visualization of structural changes Left: DeltaTreeFi1/Blank Model.emx Ancestor L	✓ Group by Delt	: flicting and Non-Conflicting :a Type a-Model Class	10del.emx
Explorer Ancestor R	i Delta Tree Co Reset Delta T	•	



Editable custom delta tree filter

Fully customizable rules, very selective suppression by element type, embedded string value, delta type, etc.

Requires no programming, but allows it!

	Move change Model Fragment Join change Model Fragment Separation change		
 Delta Tree Configuration Filters Filter out Add Changes Filter out Delete Changes Filter out Modify Changes Filter out Move Changes Filter out Fragment Join Changes Filter out Fragment Separation Changes Filter out Diagram Position and Size Changes Filter out Diagram Changes 	Type conversion change Meta-Model classes of the changed elements: Name CallBehaviorAction CallConcurrencyKind CallEvent CallOperationAction CanonicalStyle CentralBufferNode ChangeEvent Class Classifier ClassifierTemplateParameter Clause	Package uml uml uml uml uml uml uml uml uml uml	Path org.eclipse.uml2.uml.CallBehaviorAi org.eclipse.uml2.uml.CallConcurren org.eclipse.uml2.uml.CallConcurren org.eclipse.uml2.uml.CallOperation/ org.eclipse.uml2.uml.CallOperation/ org.eclipse.uml2.uml.CentralBufferf org.eclipse.uml2.uml.ChangeEvent org.eclipse.uml2.uml.Classifier org.eclipse.uml2.uml.Classifier org.eclipse.uml2.uml.Classifier org.eclipse.uml2.uml.Classifier org.eclipse.uml2.uml.Classifier org.eclipse.uml2.uml.Classifier org.eclipse.uml2.uml.Classifier
Add Remove Edit	ClearAssociationAction uml		org.eclipse.uml2.uml.ClearAssociati ⊻
Sorter: Sort by Delta Label	Search delta label string pattern Class5 (* = any string, ? = any character, \ = escape for literals: * ? \) Example: "Class1", "Package", "Fill Color", "UseCase[4-8]", "Actor[0-98[^345]]", "\[View\]", ".*Actor1.*Package" OK Cancel		

🖨 Add New Filter

Filter out condition: Match the following

My custom programmable filter

Filter name:

Type of change:

Delete change
Modify change



Reference Materials

Comparing and merging UML models in IBM Rational Software Architect – Kim Letkeman

- Part 1 Comparing models with local history
 - http://www-128.ibm.com/developerworks/rational/library/05/712_comp/
- Part 2 Merging models using "compare with each other"
 - http://www-128.ibm.com/developerworks/rational/library/05/712_comp2/
- Part 3 A deeper understanding of model merging
 - http://www-128.ibm.com/developerworks/rational/library/05/802_comp3/
- Part 4 Parallel model development with CVS
 - http://www-128.ibm.com/developerworks/rational/library/05/0809_CVS4/
- Part 5 Model management with IBM Rational ClearCase and IBM Rational Software Architect Version 7 and later
 - <u>http://www.ibm.com/developerworks/rational/library/07/0703_letkeman/</u>



Reference Materials

Comparing and merging UML models in IBM Rational Software Architect – Kim Letkeman

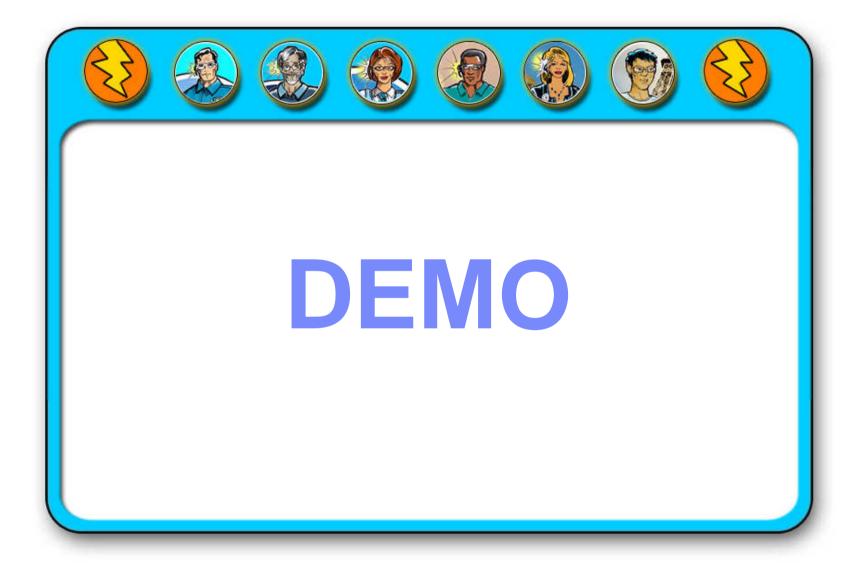
- Part 6 Parallel model development with custom profiles
 - <u>http://www-</u>
 <u>128.ibm.com/developerworks//rational/library/05/0823_Letkeman/</u>
- Part 7 Ad-hoc modeling Fusing two models with diagrams
 - <u>http://www-</u>
 <u>128.ibm.com/developerworks/rational/library/07/0410_letkeman/</u>

Related Articles

- Authoring UML profiles using Rational Software Architect and Rational Software Modeler – Dusko Misic
 - http://www.ibm.com/developerworks/rational/library/05/0906_dusko/
- Model Structure Guidelines for Rational Software Modeler and Rational Software Architect – *Bill Smith*
 - http://www.ibm.com/developerworks/rational/library/04/r-3155/

AM11

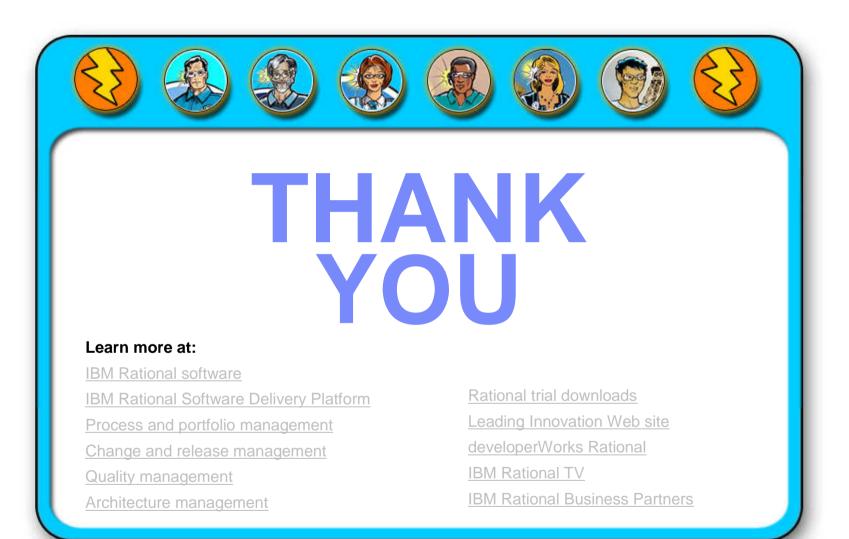












© Copyright IBM Corporation 2008. All rights reserved. The information contained in these materials is provided for informational purposes only, and is provided AS IS without warranty of any kind, express or implied. IBM shall not be responsible for any damages arising out of the use of, or otherwise related to, these materials. The information contained in these materials is intended to, nor shall have the effect of, creating any warranties or representations from IBM or its suppliers or licensors, or altering the terms and conditions of the applicable license agreement governing the use of IBM software. References in these materials to IBM products, programs, or services do not imply that they will be available in all countries in which IBM operates. Product release dates and/or capabilities referenced in these materials may change at any time at IBM's sole discretion based on market opportunities or other factors, and are not intended to be a commitment to future product or feature available in any way. IBM, the IBM logo, the on-demand business logo, Rational, the Rational logo, and other IBM products and services are trademarks of the International Business Machines Corporation, in the United States, other countries or both. Other company, product, or service names may be trademarks or service or hores.

