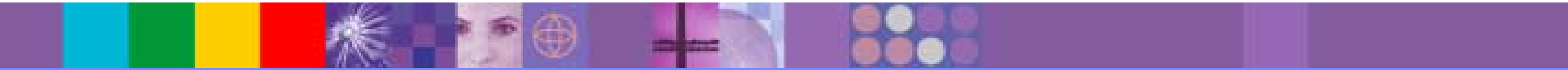
Effects of scale on methodology and tools for ADM

Lessons from experience

WebSphere software



e business on demand.

Jim Rhyne

IBM Distinguished Engineer

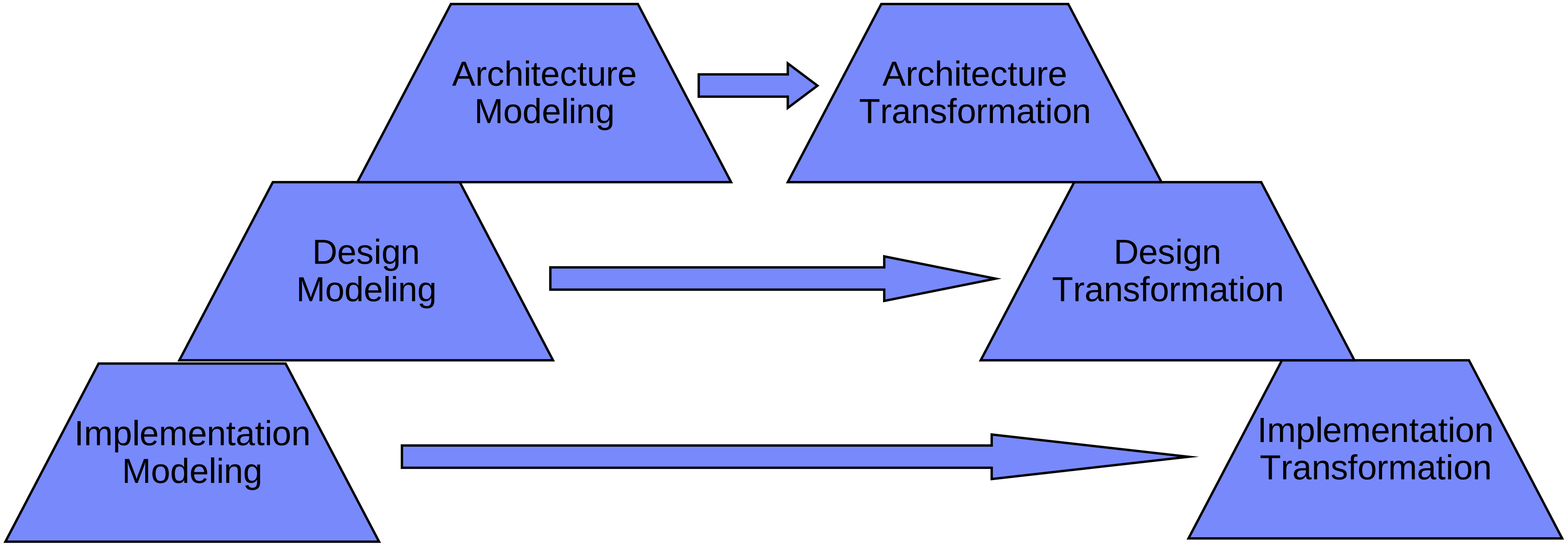
Enterprise Modernization Architect

Topics

- Methodology and tools for ADM
- Some measures of scale
- Methods for dealing with large scale in ADM analysis
- Engineering tools that scale
- Scaling requirements for metadata standards

Methodology and Tools for ADM

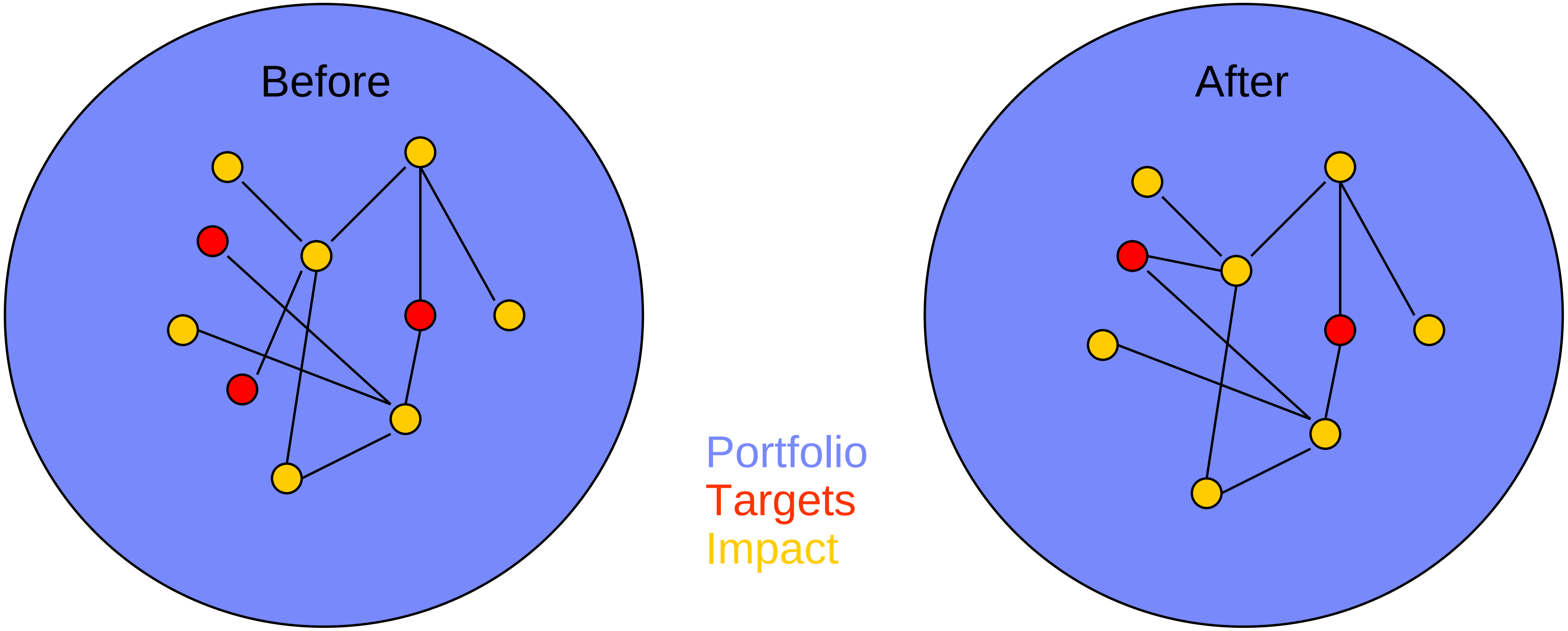
Architecture Driven Modernization for Software



Scale reduction in ADM

- Change targets have to be found
- Change impact has to be known
- Analysis tools must scale to entire portfolio

- Transformation affects small fraction of overall portfolio
- Transformation and test tools only scale to targets to be transformed



Finding things is difficult

- Relationship between business process and software architecture is locked in peoples heads
- No one person knows everything
- The people who do know something have lots of other critical work to do
  - Helping operations manage software faults
  - Planning for routine maintenance
- Recall is imperfect, things get overlooked then found at predeployment

Transforming things is also difficult

- Rule driven tools work on a few modules at a time.
- Rules are never perfect the first time. Transformations have to be backed out then improved rules applied. The longer the process goes on, the more modeuls have to be backed out and re-transformed
- Modules can be quite large.
- Implementation level metadata used by transformation rules is often 10x to 1000x larger than the source language
- Transformations must scale to team use; communication problems between individuals will cause transformational incompatibilities

Some measures of scale

Statistics from a large WSAA installation

- 90,000,000 - Individual elementary data items
- 4 months – time to complete initial artifact scan
- 600,000 – errors found in initial source scan
- 66,000 – JCL jobs
- 700 – changed modules per day

Methods for dealing with large scale in ADM analysis

Methods for controlling scale

- Details are recoverable from application sources, but architecture and business function are not
  - Architecture and business models need to be created, recorded and maintained
  - But, the architecture and business function needs to be coupled to the sources to allow recovery of detail.
- Analysis proceeds from top (i.e. business function and architecture) down
  - Analysis work can be delegated to additional analysts as work proceeds
  - High level models must be available to support top down approach
- Transformation strategies are elaborated as analysis proceeds
  - Transformation strategy defines additional analysis work to be done
  - Supports early back out when transformation strategy appears likely to fail

More methods for controlling scale

- Analysis work benefits from planning, coordination, and sharing
  - Analysis work can be planned in a hierarchical manner
  - Key aspects of analysis should be done early to facilitate later, more detailed work
  - Sharing analysis work avoids duplication, provides consistency
- Analysis methods should identify high risk, high cost items early in the analysis
  - Can provide early warning of budget and schedule overruns
  - Can also provide alternate transformation strategies to reduce cost and risk
- Use tools instead of manual methods
  - Tools scale better
  - Tools have perfect memory
  - Tools do not make mistakes (but heuristic, conservative analysis algorithms produce more data for human review)

Useful tools for analysis

- Record “as built” software at executable, dataset, deployment level of detail
  - Manageable level of detail for preliminary analysis
  - Used to identify candidates for more detailed analysis
- Record linkages between Entity/Activity model of business processes and “as built” model of supporting software
  - Allows top down analysis of change driven by business process change
  - Transformation must change business process model as software and other models are transformed
- Record hardware, communication, OS topologies and linkages to “as built” software
  - Allows top down analysis of change driven by technology and operations change
  - Topology model and linkages must be transformed as software and other models are transformed

Additional useful tools for analysis

- Record analysis project results for reuse, but eliminate when no longer needed
  - Save storage space and reduce working set size for analysis
- Tools for planning analysis projects
  - Initial analysis results in concrete results plus conjectures for additional analysis needing to be done.
  - Record these conjectures
    - So they do not get lost
    - So they can be handed out to others for completion
  - Record linkages between conjectures and concrete analysis results
    - Avoid redoing analysis that has already been done
    - Can be used to eliminate analysis data no longer needed to resolve conjectures

Designing analysis tools that scale

Support for top down analysis

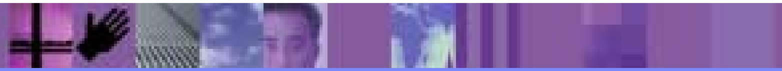
- Analysis algorithms and data schemas must support performing additional work when analysis details are not found.
  - "Open vs. closed universe assumption"
    - In "closed universe", null means negation
    - In "open universe", "don't know" and "negation" have distinct values
  - Plans for resolving "don't know" conjectures must be effective and tailorable
  - Plans for resolving "don't know" conjectures typically require:
    - Work list processing (e.g. first do X, then if the results show Y is true, do Z)
    - Backtracking (Z failed, so undo X and select another plan
    - Recording successful and failed plan executions (so analysts can understand what happened that produced the current results)
  - Plan application criteria consist of pattern match scores on "open universe" data models
    - Scores determine likelihood of plan success, sort plan applications to avoid wasting time

Use transactional, relational datastores

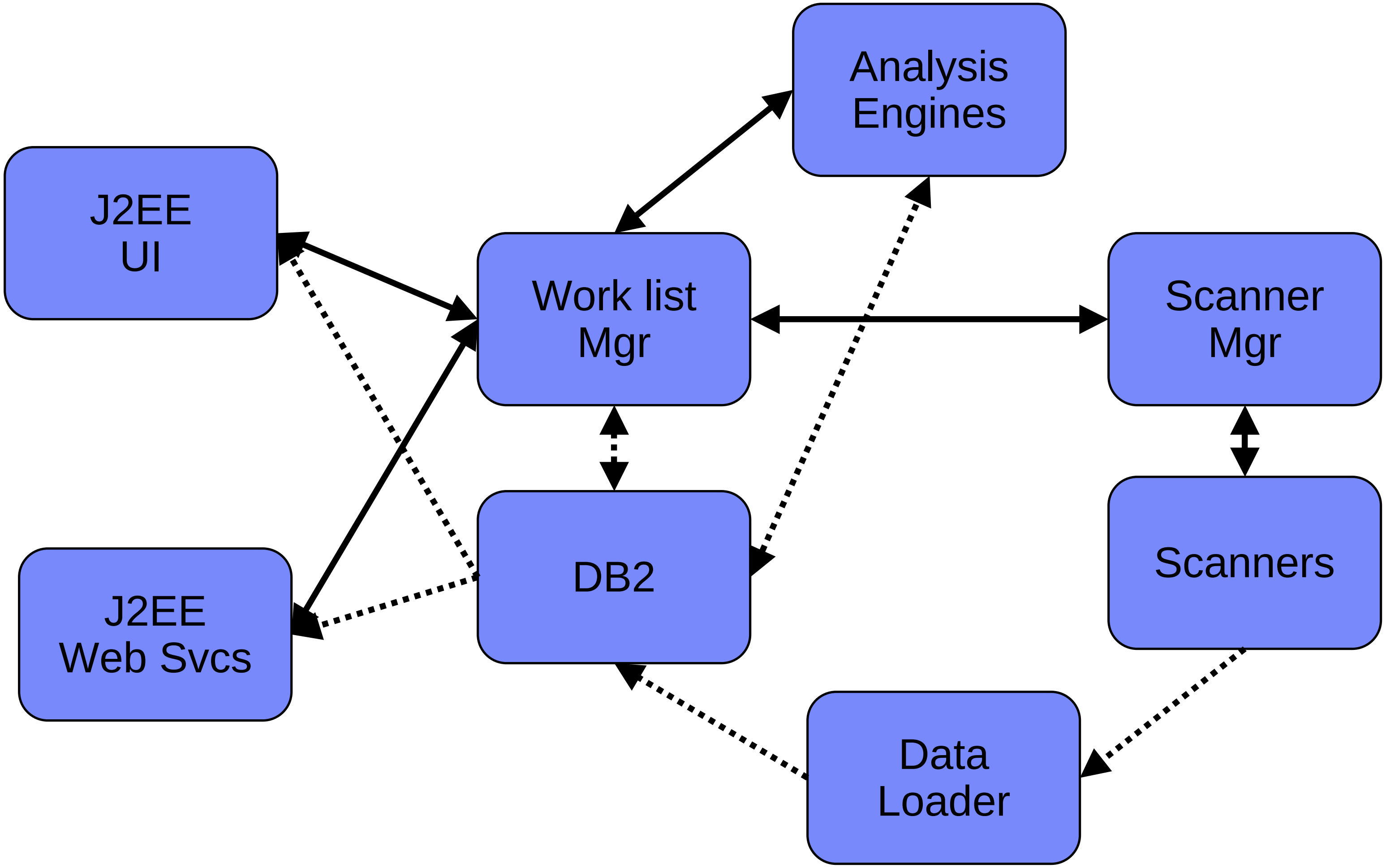
- Metadata changes with the application
  - Data inconsistency causes analysis algorithm failure
  - Metadata changes are input from multiple, asynchronous sources
- Careful design of relational schema is required
  - 3rd Normal Form schemas do not perform (multi-way joins)
  - Query results may need to be cached for performance (tradeoff space for speed)

Design User Interface for scale

- Selection lists are not practical for 50,000 items
  - Use search instead
  - Remember recently used search keys and analyses
- Design UI for drill down
- Disconnect UI thread from analysis threads
  - Avoid UI lockup during long queries
- Advise user about the time and cost of prospective analyses
- Allow analysis in background
- Graphical displays of more than 40-50 items are useless
- Mark and remember things of interest seen during browsing
  - Click to return to them
- Eliminate duplicate or derivative error indications
  - Reduce amount of manual work to be done.



Websphere Studio Asset Analyzer Architecture



Scaling requirements for metadata standards

Scale recommendations for ADM standards

- Metadata model should be segmented
  - Avoid the need to populate the entire model when only part is needed
- Assume “open universe”
  - Define “don’t know” values
  - Provisions for recording hints to resolving “don’t know” conditions
- Associate timestamps and source identifier with metadata
  - Allows validation of metadata
  - Allows purging of unneeded out outdated data
- Design for retention of hand added data across metadata updates

Conclusions

- Scalability of ADM tools is important and can be achieved
  - Flexible metadata models allow space/time tradeoffs
  - Transactional, relational metadata stores allow change with data consistency
  - Server architecture supports team use
  - Automated analysis planning supports recovery of needed metadata with minimum human labor
- ADM metadata standards allow diverse tools to interact to help ADM transformations
  - Segmentable and extensible standards are needed
  - Support “open universe” assumption