

## View Point



### Reuse in EAI Software

#### A Practitioner's Perspective

---

Rakesh Kumar Mishra

#### The Challenge

Reuse is one of those IT best practices that are quite simple to understand and appreciate but very difficult to implement on the ground. With rising costs of software development in unfavorable economy, organizations have started pushing very hard on 'real' reuse. This 'real' reuse not only demands significant cost saving but also requires sustainability of the reuse.

There are mixed schools of thoughts as far as 'realness' of the reuse is concerned. One school of thought is highly supportive of the reuse philosophy and believes in investing into processes and infrastructure necessary for reuse adoption. Propagators of the other school of thought believe that reuse is utopian concept and should only be considered for investment in highly tactical or opportunistic manner. This in particular becomes very challenging when both the schools of thoughts are prominently present in the same organization.

Infosys experience suggests that either of the extreme view-points is not suitable for sustainable 'realness' of the reuse. Infosys believes that a balanced approach will help organization to build a long term foundation for reuse. This balanced approach supports tactical / opportunistic view as well as recommends investing selectively into the foundation for reuse.

This view-point paper will provide clear insight into aspects of reuse for EAI software development and share the Infosys experience on quantitative reuse benefits.

## Essence of the Infosys Experience with Reuse

Organizational learning is the most valuable knowledge in the area of reuse because of its people centric nature. Essence of the decades of Infosys' experience in reuse can be summarized as follows:

1. **Reuse is not about just the cost savings** – while cost savings are the main driver for organization to go after reuse, that is not all and actually that should not be only decision making criteria. Key objectives of reuse in software development should be:
  - a. Improve productivity and hence reduce the cost of development
  - b. Improve the use of standardized components in the solution and by doing that improve the product quality and long term manageability
  - c. Simplify the development process by reducing the need to develop the components from scratch
2. **Discipline is the most important factor** – Making reuse work in the organization requires high degree of discipline at an individual level. Many organizations have failed in implementing reuse despite having great systems and processes simply because of the poor acceptance of the discipline required. Discipline is reasonably difficult issue to resolve and one of the ways it can be addressed to some extent is by incentivizing people when they reuse.
3. **Reuse Infrastructure maturity** – Since discipline is the biggest issue, availability of a good reuse infrastructure goes long way in making it easy for people to search, discover, access and use the assets. More complex the processes and systems are, lesser the chances are that people will adopt it. Reuse infrastructure includes repository of assets, access channel (e.g., a portal), configuration control system, reuse service request management tool etc.
4. **Metrics of effectiveness** – There are metrics associated with reuse that are direct reflection of the effectiveness of the reuse, for example total effort saved because of reused assets. Very often these are not adequately measured or measured in inappropriately manner that results into wrong data-points and hence wrong perception about reuse.
5. **Build architecture blueprints that enable reuse** – another way of reducing the level of discretion required at project level about reuse is to have the standard architecture and design blueprints support reusable frameworks and components. That to some degree enforces the development team to have the appropriate reuse considerations while designing the solutions.
6. **Invest into reusable common components** – Many organizations don't invest into reuse foundation infrastructure and expect their teams to deliver reuse. Reuse is highly unlikely to give meaningful results if it is based on completely 'adhoc' and 'unplanned' methods.

## The Reuse Foundation Map

Reuse foundation map provides basic understanding of elements in the reuse model. This is important for planning reuse strategies and investments.

Reuse Category	Reuse Items	Main Benefits Types	Level of investment and effort required
Software Meta Components	<ul style="list-style-type: none"> <li>• Schemas</li> <li>• Configuration meta-data</li> <li>• Design templates</li> <li>• Code templates</li> <li>• Reference data</li> <li>• Data models</li> <li>• Mapping rules</li> <li>• Etc.</li> </ul>	<ul style="list-style-type: none"> <li>• Effort savings</li> </ul>	Low
Software executable components	<ul style="list-style-type: none"> <li>• Functional services</li> <li>• Technical Services</li> <li>• Technical Code</li> <li>• Tools/utilities</li> </ul>	<ul style="list-style-type: none"> <li>• Effort savings</li> <li>• Product quality</li> </ul>	High
Process Assets	<ul style="list-style-type: none"> <li>• Guidelines</li> <li>• Checklists</li> <li>• Methodologies</li> <li>• Templates</li> </ul>	<ul style="list-style-type: none"> <li>• Efficiency</li> <li>• Product quality</li> </ul>	Moderate
Knowledge Assets	<ul style="list-style-type: none"> <li>• Best practices</li> <li>• Body of knowledge</li> <li>• Experiential learning database</li> <li>• Etc.</li> </ul>	<ul style="list-style-type: none"> <li>• Reduced cost of quality</li> <li>• Reduced cost of failure</li> <li>• Effectiveness of the solution</li> </ul>	Moderate

## Quantitative Reuse Benefits

Based on the Infosys experience of reuse, especially in EAI development, here are the realistic projections about the reuse benefits in terms of cost savings that organization can expect:

Reuse Category	Savings Projection	Realization period
Software Meta Components	<b>Short term:</b> 3-5 % of the design and build effort <b>Long term:</b> 8-10 % of the design and build effort	6 months+
Software executable components	<b>Short term:</b> 5-8 % of the overall development effort <b>Long term:</b> 15-25% of the overall development effort	8 months+
Process Assets	<b>Short term:</b> 2-3 % of the overall development effort <b>Long term:</b> 8-10 % of the overall development effort	6 months+
Knowledge Assets	<b>Short term:</b> 1-2 % of the overall development effort <b>Long term:</b> 8-10 % of the overall development effort	12 months+

### About the Author

**Rakesh Mishra** is a senior principal consultant with Infosys Limited with more than 11 years of experience in Integration. Rakesh specializes in Integration Strategies and Integration Competency Centers and has helped various fortune 500 companies to make right technology and architecture decisions to set up enterprise-wide integration strategies that deliver business value from integration.



For more information, contact [askus@infosys.com](mailto:askus@infosys.com)

### About Infosys

Many of the world's most successful organizations rely on Infosys to deliver measurable business value. Infosys provides business consulting, technology, engineering and outsourcing services to help clients in over 30 countries build tomorrow's enterprise.

For more information about Infosys (NASDAQ:INFY), visit [www.infosys.com](http://www.infosys.com).