

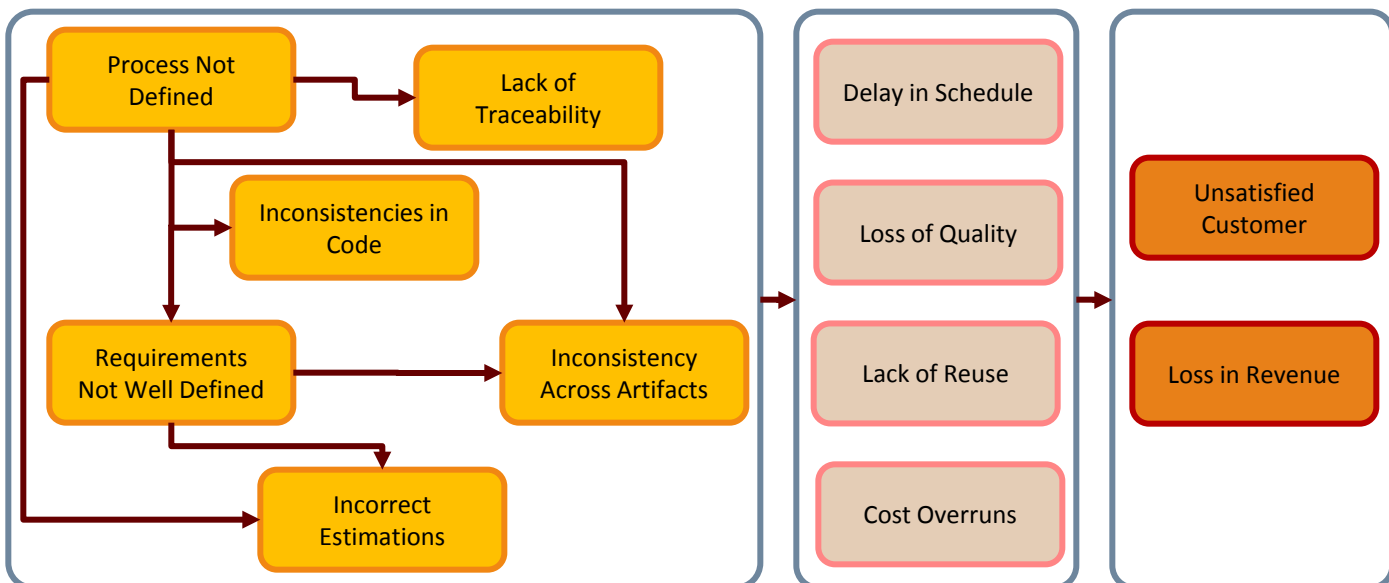
Software Development Automation Tool

PRODUCT DATA SHEET

Common Problems in the Software Development Process →

Software projects that suffer from poor quality, schedule delays and cost overruns are often found to have one or more of the following problems:

- **Development processes are not defined or not followed:** Defining an efficient development process is critical to the success of any IT organization. Even after development processes are defined, project stakeholders often fail to follow a process due to a lack of knowledge about the process or because the process itself is difficult or time consuming.
- **Requirements are not defined in sufficient detail:** In a lot of projects requirements are defined in plain text format. Standard modeling techniques are not used often enough and this leads to a lack of clarity in the requirements.
- **Incorrect estimations:** Lack of a proper estimation process results in estimations done in an ad-hoc manner. This frequently results in projects suffering from schedule delays. Furthermore the pressure of schedule delays results in hurried implementation of functionality that is low on quality.
- **Inconsistency of content in artifacts across various phases:** If design documents are inconsistent with requirements or if implementation is inconsistent with requirements or design, the final product will deviate from the original requirements.
- **Lack of reuse across or within systems:** Even if similar systems or functionalities exist, it is not possible to identify similarities or analysis itself is so time consuming that reuse is not possible across systems and this results in duplication of functionalities.
- **Inconsistencies in code:** Code written by different developers often deviates from platform and framework standards. Such code is prone to bugs and is difficult to manage.
- **Lack of traceability:** Impact analysis during change management is a critical activity in any software project. Lack of structured information makes it difficult to trace the impact of a change leading to incomplete or incorrect analysis. As a result, systems become less maintainable.

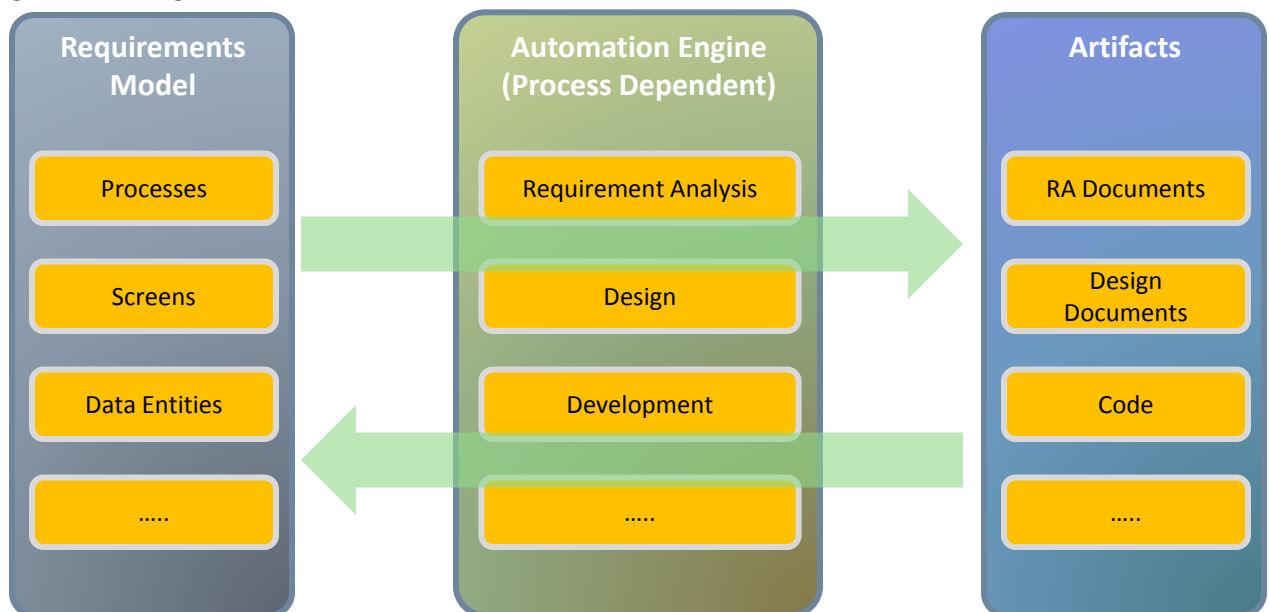


These problems cause schedule delays and bring about an overall reduction in software quality. This in turn leads not only to reduced profits but also to unsatisfied customers. So addressing these problems are critical to the growth of any IT organization.

Improving Software Development Process With Modeling & Automation →

One of the solutions to tackling the above mentioned problems is to increase automation across processes. Automation of process artifacts reduces time and, if done properly, improves quality and consistency of content. In order to automate processes the following must be done:

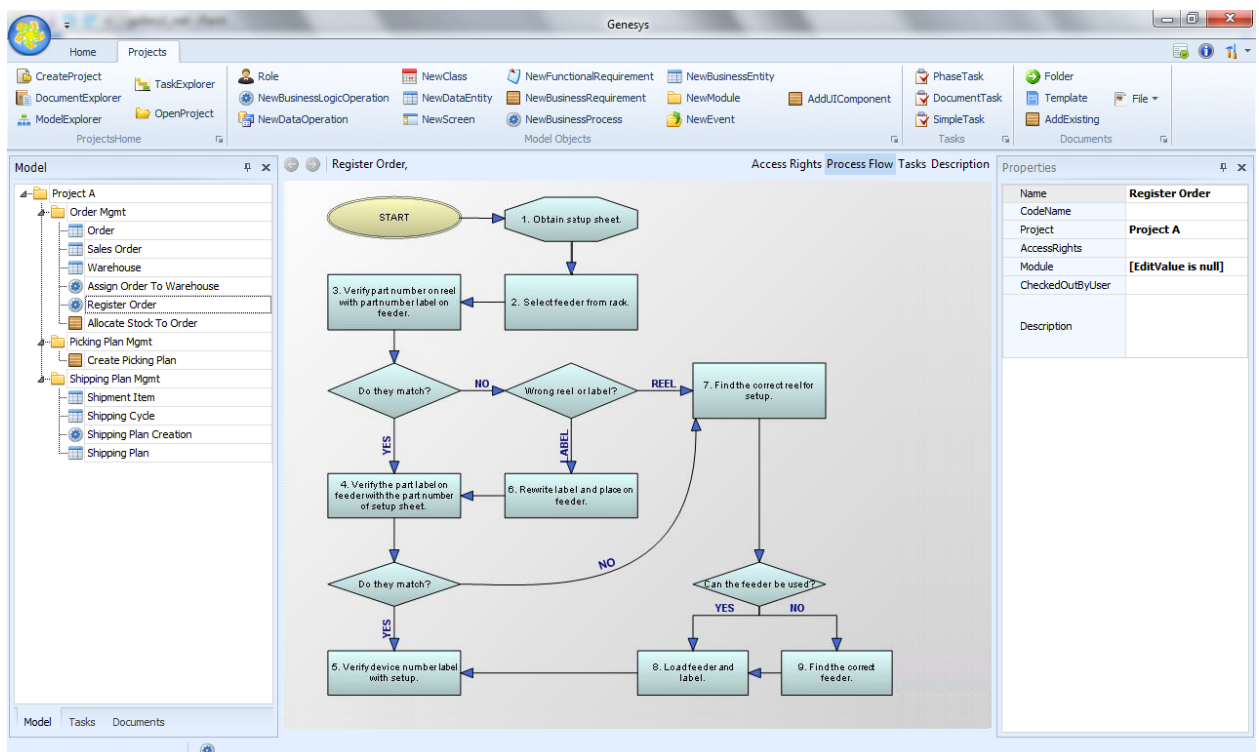
- **Define efficient processes:** Development processes need to be well defined. They should be easy to follow. Artifacts for each activity of a process need to be well defined. Well defined artifacts will make automation easier.
- **Proper requirement modeling:** A formal modeling language or methodology should be used which allows structured definition of requirements. A structured definition of requirements makes content easier to access for automation purposes and it also reduces redundancy facilitating reuse from the requirements phase itself.
- **Using an estimation methodology:** Well defined estimation methodologies like “Function Point Analysis” or “COCOMO” ensure consistency in estimates. Repeated use of an estimation methodology allows an organization to refine estimation parameters. This results in increased accuracy of estimates. Automation of estimation using high level requirements will ensure completeness of the estimate. Automation will also make the estimation process less time consuming and consistent.
- **Automation of project planning processes:** Project management artifacts like project schedule should be derived directly from the high level requirement model. Automating this process will ensure completeness and consistency.
- **Automation of design and code:** Design can be automated from the requirements model. Well defined and structured requirements will make generating a skeletal design easier. Such a design can later be detailed out manually. Design, once detailed out, can be used to generate code. Code generation ensures better quality and standardized code according to the platform and framework.
- **Defining framework dependant automation processes:** Automation of design and code should be done for every framework being used for development. It should be possible to use the same requirement model to generate design and code for different framework.



Automation With Genesys →

Genesys is a software development automation tool based on modeling of system requirements and design. Generation of artifacts across phases can be done using this model. Existing artifacts can also be used to reverse engineer the requirement model. Genesys can be used as:

- **A modeling tool:** Genesys is used to model requirements and design. In Genesys, a model is described in terms of 'Types' such as business processes, functional processes, screens, events, database queries, etc. The types used for modeling purposes can be customized up to the attribute level. Customized designers can be plugged in for any given type.
- **A process driven automation tool:** Development processes can be defined and artifact templates for a process can be plugged in. The 'types' of the model applicable to a process can be specified. Transformers that transform the specification model to implementation model or model to artifacts, or vice versa, can also be plugged in for customized automation.
- **A code generation tool:** Frameworks for web, mobile and desktop application on platforms such as .Net, Java, PHP, can be defined as processes and code generators for every framework can be plugged in. Code generation can be done for any framework from the same requirement model.
- **A traceability tool:** When artifacts are automated from the model, links of these artifacts are maintained with the model using which they were created. These links help to trace the impact of a change made to the model, or a change made to an artifact.

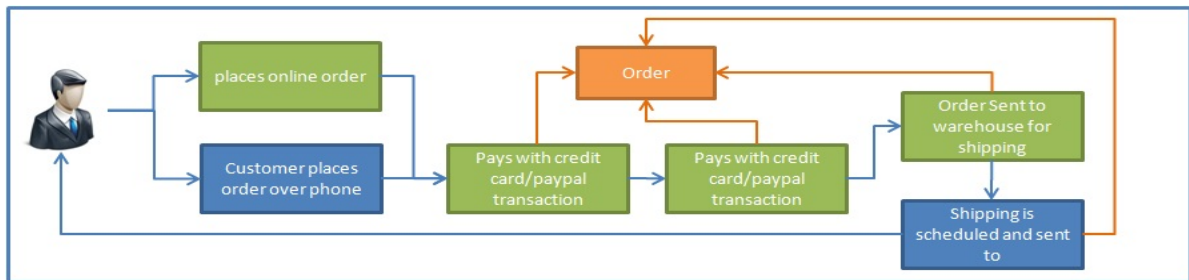


Modeling Types Supported By Genesys →

The following types are defined in the Genesys model. These types can be customized at the attribute level. New types can also be added as per process or project requirements.

Requirement Analysis Model →

- **Business Process:** Business Processes are defined in form of process flow diagrams and can contain functional processes, screens, business entities and roles.

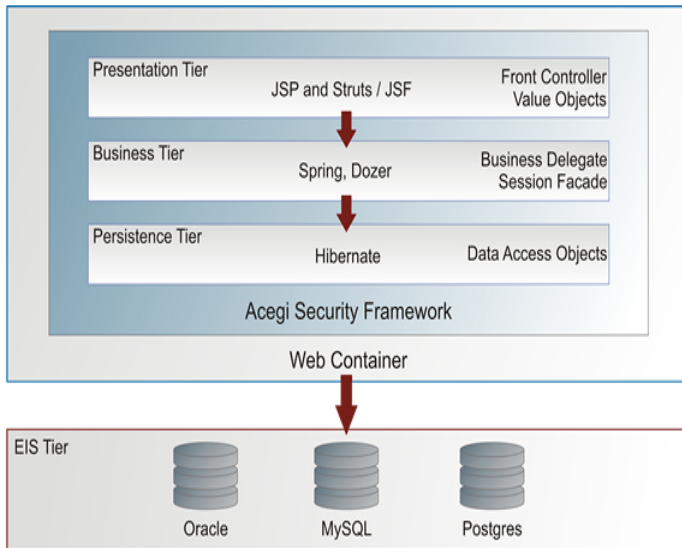


- **Business Entity:** These define the data model as defined at the requirement analysis level.
- **Business Rule:** Business rules of the system. These rules may be associated with business or functional processes or business entities.
- **Role:** Roles of users of the system
- **Non-Functional Requirement:** Defined in plain text format.

Design Model →

- **Functional Process:** These are first defined during requirement analysis and detailed out later during the design phase. Functional Processes are defined in form of process flow diagrams and can contain screens, operations and data entities.
- **Data Entity and Data Relation:** Design level detailing of business entities. Entity attributes and relations between entities are specified in detail.
- **Operations (Application Logic, Business Logic, Persistence Logic):** Application logic and business logic operations are defined in terms of input and output only. Further detailing out is done at the code level itself. Database operations are defined in terms of SQLs.
- **Screens and UI Components:** Screens are first defined during requirement analysis and detailed out later during the design phase. Screen fields and event and event handlers(operations) are detailed out during design.
- **Events:** System level events for batch processes or events occurring in screen operations.
- **Database schema:** Database tables, columns and constraints. This can be derived from data entities or vice versa.
- **Class Diagram:** Classes can be defined. Class diagrams can be created(Future enhancement).

Frameworks Supported By Genesys →



J2EE Framework With JSF, Spring & Hibernate

A J2EE application framework for developing web robust applications. The framework uses JSF for presentation and Hibernate for persistence. Spring is used for transaction management and bean lifecycle management. Genesys generates end-to-end code for applications developed on this framework.

Future Enhancements Planned →

Automation of Project Planning Artifacts

Estimates and schedules can be generated from high level requirements.

Code Generation for ASP.NET Applications

End-to-end code for ASP.NET applications will be generated from the specified model.

Code Generation for Applications on Mobile Platforms (Android, iOS, etc.)

End-to-end code for mobile applications on Android, iOS, etc. from the specified model.

System Requirements →

Hardware

Intel Core 2 Duo-class Processor, 1.0 GHz and above
Recommended → Intel Core 2 Duo, 2.0 GHz

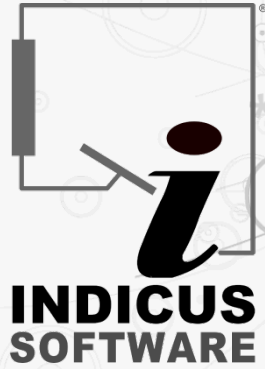
512MB RAM and above
Recommended → 1024MB

More than 2 GB of disk space

Software

NET Framework 3.5

IIS 5.0 and above



Software Development Automation Tool

Indicus Software

28, Varshananda Society
Anandnagar, Sinhagad Road
Pune 411-051, Maharashtra, India
Tel: +91-20-2434-1287/88
FAX: +91-20-2434-1289

Indicus Software Japan

Kyodo Building, 5F-53
Nihonbashi-Kakigaracho 1-3-5
Chuo-Ku, Tokyo 103-0014
Tel: +81-3-6206-2945
FAX: +81-3-6206-2946

 info@indicussoftware.com

 <http://www.indicussoftware.com/>