Requirements Engineering in the bio medical industry: GAMP 5 and tooling

Gauthier Fanmuy
ADN
http://www.adneurope.com
39-41 rue Louis Blanc
92038 Paris La Defense France
gauthier.fanmuy@adn.fr

Richard Szczepaniak
CORTIM
http://www.cortim.com
49 bis av des Pyrénées
31120 Lacroix Falgarde France
richard.szczepaniak@cortim.com

Abstract: The principles of systems engineering are applicable to all types of systems including computerized systems. They are particularly effective for these systems because the computer is still new technology, they can better understand the development and life cycle of these systems by erasing the image of "black boxes". It is now essential for computer systems to carry more and more critical actions especially in highly regulated industries such as pharmaceuticals or medical devices industries. These industries has established a repository, the GAMP (see reference [1]), in order to define the correct principles to be established for computerized systems. This paper presents the principles of systems engineering defined in GAMP and how they were developed by the company ADN and CORTIM for the pharmaceutical and medical devices industries.

BACKGROUND

For twenty years, the development of IT systems in the industry has put forward new problems during installation and use of these systems:
- Generic systems with multiple users
- Global and multi-sites
- Technology systems become more complex and often uncontrolled, or partially, by the end users
- Maintenance and monitoring of changes in systems

These points become critical when applied to the highly regulated industries such as pharmaceuticals, where legislation demands thorough controls (GMP, GMP, 21 CFR Part11, 21 CFR Part820, see References [2], [3] and [4]) and IT systems support to medical devices. To address these issues, the pharmaceutical industry through ISPE has developed a good practice guide for the development and operation of computerized systems: the GAMP whose version 5 was released in 2008.

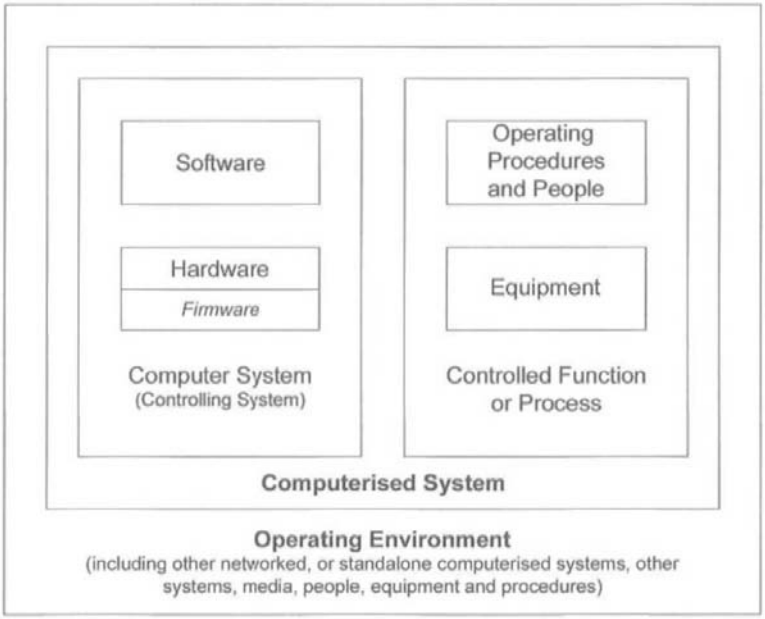
Version 5 of GAMP explicitly included the key principles of systems engineering and on this, the ADN Company, leader for 15 years in validation and compliance of computerized systems in the pharmaceutical and medical devices industries, has developed and offers to its customers the tools to implement the methodology recommended by the GAMP with Systems

Engineering good practices. This choice was confirmed by the convergence of GAMP 5 in different benchmarks with other repositories such as ITIL, CMMI and with ISO standards (ISO 15288, ISO62304).

GAMP is not a regulation but a collection of good practices. But its implementation in the development of computerized systems guarantees the compliance to the FDA (Federal Drug and Administration) regulations.

Requirements Engineering in the GAMP 5

The first System Engineering principle shown in GAMP 5 is modeling a computerized system into hardware, software, and network components together with the controlled functions and associated documentation. This covers a broad range of systems from document management systems to clinical trials data management systems.



PIC/S Good Practices for Computerised Systems in Regulated "GXP" Environments' (PI 011).

Figure 1 : Modeling of a computerized system according to GAMP 5

Also is considered is the life cycle of the system globally.

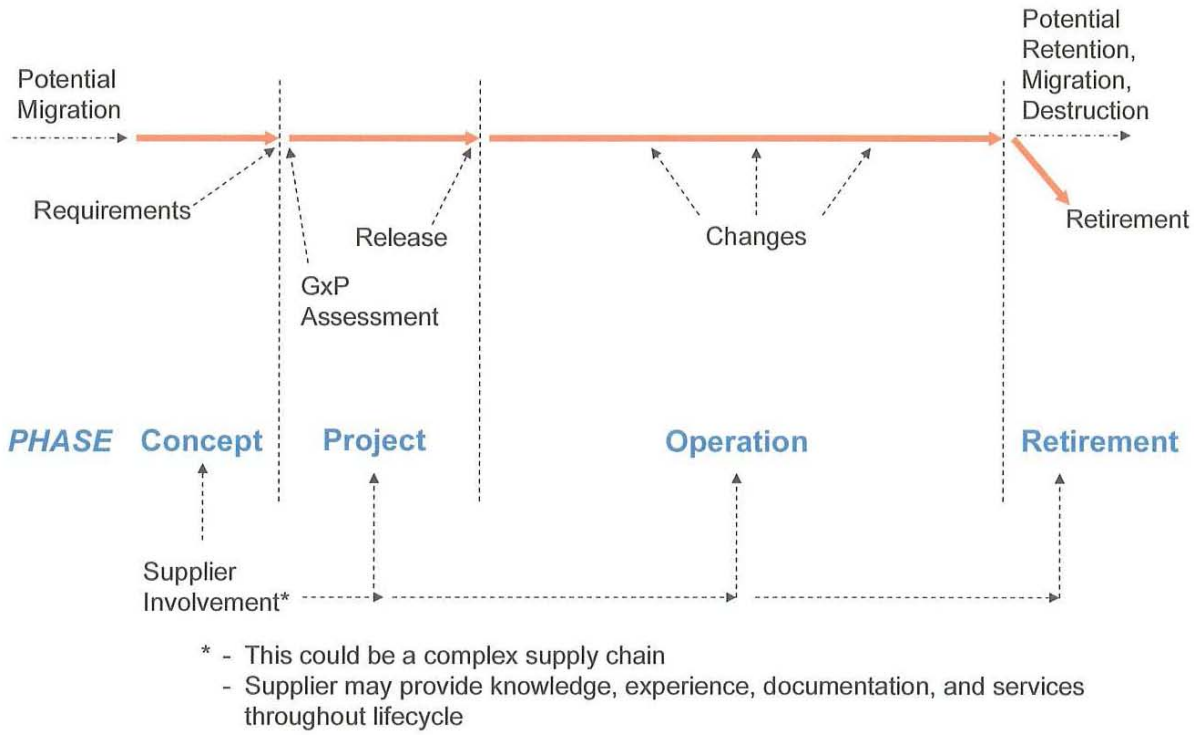


Figure 2 : Life cycle of a computerized system according to GAMP 5

These two points allow a better understanding of the computerized system at first instance by separating the hardware and software parts of the system but also taking into account the entire life cycle of the system. The system is no longer a black box and its life cycle is not limited to the operation.

Throughout this lifecycle, we will set out systematically to reproduce each evolution of the system. And this from the design of the system until it is withdrawn
Throughout this lifecycle, approach entails activities are defined and performed in a systematic way for conception, understanding the requirements, release, and operational use, to system retirement.

The first phase is called **Concept.** During this phase discussion on the future system is initialized: the needs, the budget anticipation, the risk-benefit balance, a preliminary study of solutions.
The second phase is the **Project**. It relates to the activities of management and planning, specification, configuration and coding at different levels of abstraction (systems, hardware, software...), assessment and selection of providers, verification, reporting and release.
The risk management is applied to identify and reduce risks to an acceptable level.
The third phase called **Operation** is often the longest phase. It is managed by defined and maintained procedures, and applied by trained personnel with education and appropriate experience. The key is maintaining control of the system's ability to meet the needs, and regulatory compliance.
The final phase is the **Retirement** of the system, whose main issue is security and continuity of data processed by the removed system.

The second System Engineering principle in GAMP 5 is Requirements Management. It uses support processes such as: risk management, traceability, design reviews, management of change and configuration management documentation.

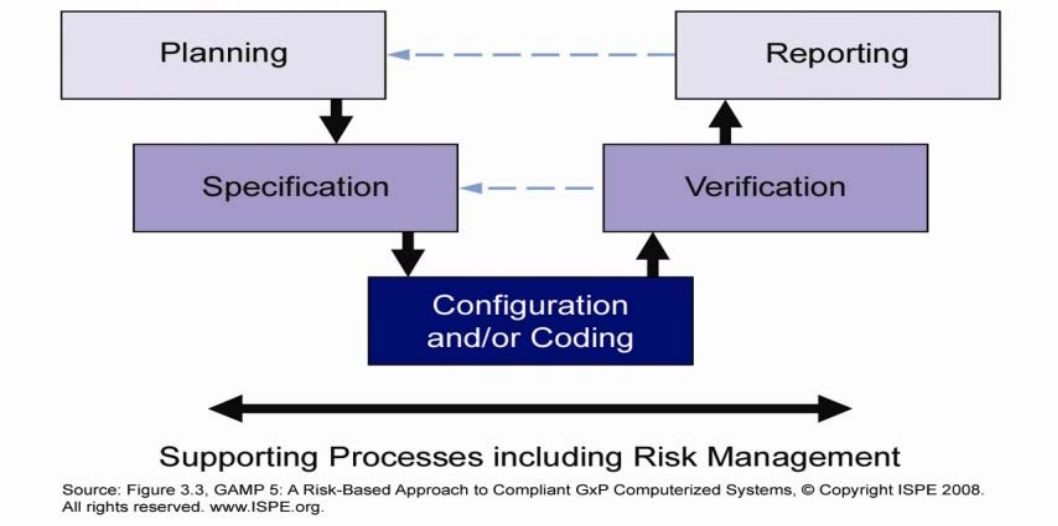


Figure 3 : Process activities

Requirements Management is a systematic activity which deliverables are defined according to different categories of systems.

Category	Description	Typical Examples	Typical Approach
1. Infrastructure Software	• Layered software (i.e., upon which applications are built) • Software used to manage the operating environment	• Operating Systems • Database Engines • Middleware • Programming languages • Statistical packages • Spreadsheets • Network monitoring tools • Scheduling tools • Version control tools	• Record version number, verify correct installation by following approved installation procedures • See the *GAMP Good Practice Guide: IT Infrastructure Control and Compliance*
3. Non-Configured	Run-time parameters may be entered and stored, but the software cannot be configured to suit the business process	• Firmware-based applications • COTS software • Instruments (See the *GAMP Good Practice Guide: Validation of Laboratory Computerized Systems* for further guidance)	' Abbreviated life cycle approach • URS • Risk-based approach to supplier assessment • Record version number, verify correct installation • Risk-based tests against requirements as dictated by use (for simple systems regular calibration may substitute for testing) • Procedures in place for maintaining compliance and fitness for intended use
4. Configured	Software, often very complex, that can be configured by the user to meet the specific needs of the user's business process. Software code is not altered.	• LIMS • Data acquisition systems • SCADA • ERP • MRPII • Clinical Trial monitoring • DCS • ADR Reporting • CDS • EDMS • Building Management Systems • CRM • Spreadsheets • Simple Human Machine Interfaces (HMI) Note: specific examples of the above system types may contain substantial custom elements	• Life cycle approach • Risk-based approach to supplier assessment • Demonstrate supplier has adequate QMS • Some life cycle documentation retained only by supplier (e.g., Design Specifications) • Record version number, verify correct installation • Risk-based testing to demonstrate application works as designed in a test environment • Risk-based testing to demonstrate application works as designed within the business process • Procedures in place for maintaining compliance and fitness for intended use • Procedures in place for managing data
5. Custom	Software custom designed and coded to suit the business process.	Varies, but includes: • Internally and externally developed IT applications • Internally and externally developed process control applications • Custom ladder logic • Custom firmware • Spreadsheets (macro)	Same as for configurable, plus: • More rigorous supplier assessment, with possible supplier audit • Possession of full life cycle documentation (FS, DS, structural testing, etc.) • Design and source code review

Figure 4 : The different categories of computerized systems according to GAMP 5

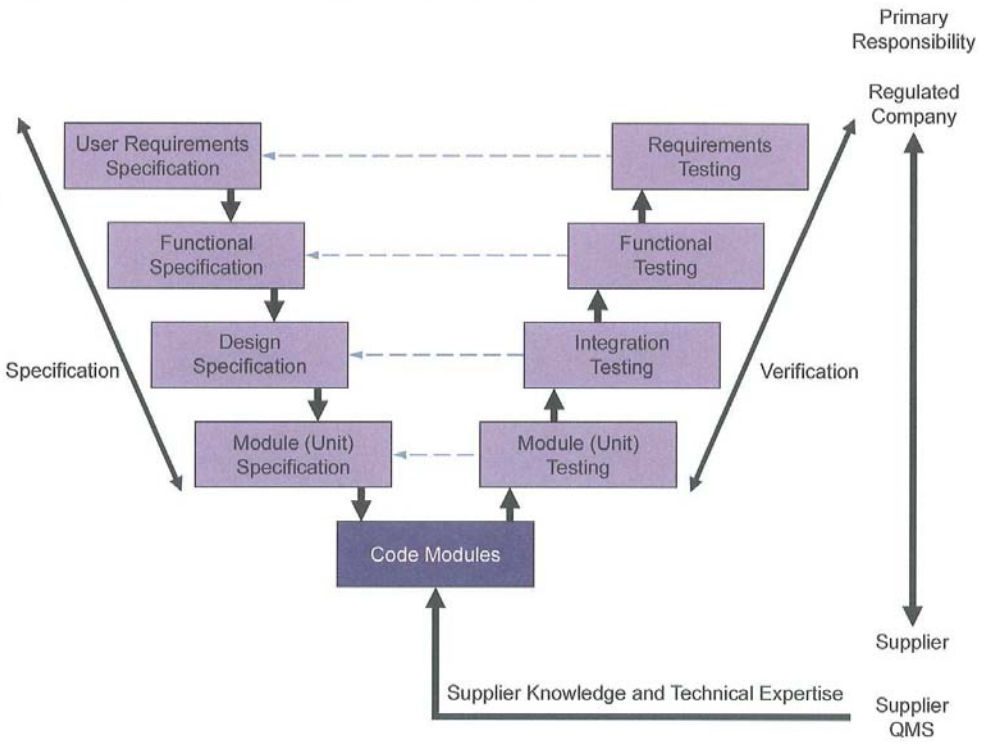


Figure 5 : Approaching a computerized system in category 5 according to GAMP 5

Figure 5 shows an example of a custom system: the specification of deliverables for this system are the URS (User Requirements Specification: describes the needs), FS (Functional Specification: describes Technical Requirements in response to needs), DS (Design Specification: describes the technical solution in response to Technical Requirements).
Figure 5 also highlights one of the key Requirements Engineering support process: Traceability. Traceability is not limited to developing traceability matrices early phase operation as proof of regulatory compliance, but as a true engineering tool for ensuring that the needs and regulation constraints are taken into account into the development process, that an exhaustive risk based approach has been done, that the computerized system is correctly verified and validated.

But the GAMP 5 goes further and also defines outlines for the different types of specifications, and rules for quality requirements: SMART

- S: Specific (single)
- M: Measurable (measurable)
- A: Achievable (feasible)
- R: Realistic (Realistic)
- T: testable (testable)

Different types of requirements are considered:

- Operational requirements
- Functional requirements
- Data requirements
- Technical requirements
- Interface requirements

- Environment requirements
- Performance requirements
- Availability requirements
- Security requirements
- Maintenance requirements
- Regulatory requirements
- Migration of any electronic data
- Constraints to be observed
- Life cycle requirements

Change management and configuration processes are also engineering support processes. The management changes, for example, confined in GAMP version 4 to operation phase are extended to the entire life cycle.

Finally GAMP 5 proposes streamlining testing efforts based on risk analysis. It helps to identify possible failures of a system and to rate their impact vis-à-vis patient safety, and to measure the testing effort required. Note that the GAMP 5 advocates making iterative risk analysis in order to add new requirements to reduce the impact of failures identified.
Risk analysis is conducted gradually. It starts in the upstream phase on the business processes involving the computerized system and the definition of the stakeholder needs (URS). It continues parallel of the functional specification (FS) and design specifications (DS) by a risk analysis at function and requirement level.

The methodology developed by ADN

Main gaps in projects (delays, quality, and costs) have multiple causes (see Figure 5):

- Poor requirements definition and control
- Insufficient project or engineering management
- Lack of data management
- Deficient management of complexity

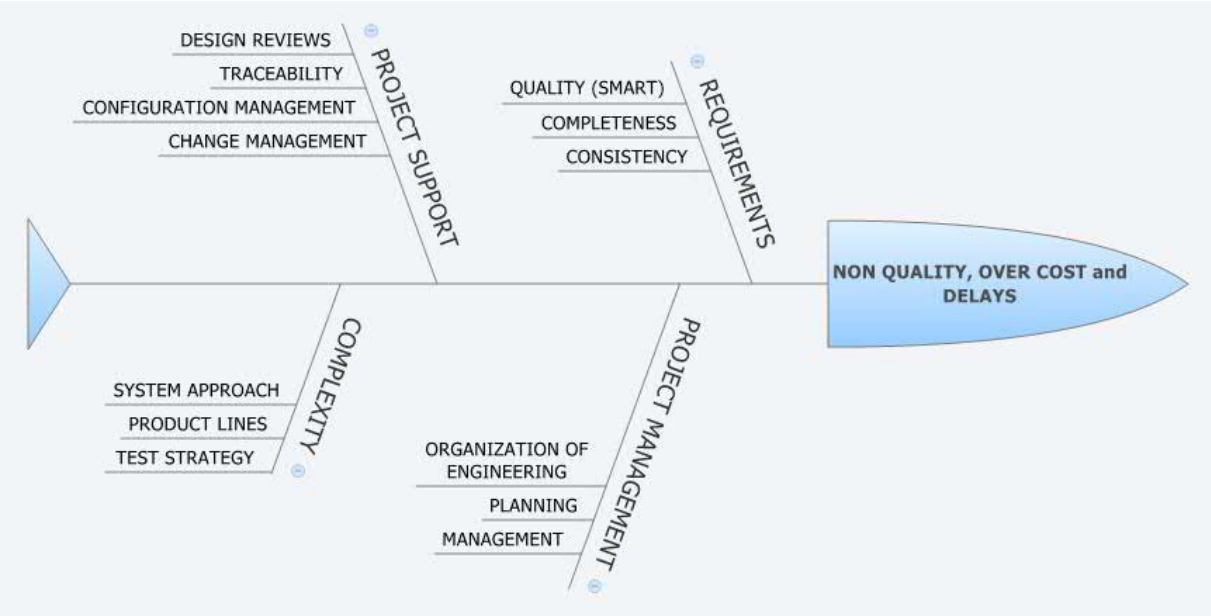


Figure 6 : Typical source of gaps in projects

To address these gaps, strengthened in GAMP 5, we have developed or integrated some add-ins in 2 tools:

- DOORS: IBM software for requirements management and traceability

  DOORS is a leading tool on requirements engineering: DOORS project consists of different modules where each object can be linked with another object in the same module or not.

- Quality Center: Hewlett Packard software for requirements, risks, test management and traceability.
  Quality Center is a leading tool for testing, since version 9 it includes requirements and risks.

We introduce in this paper some of the major developments made in these 2 tools by considering 2 aspects:

- Requirements Quality: SMART
- Traceability: Data model
- Electronic signature

Requirements – Quality issues (SMART):

If we consider the SMART properties, some of the properties of requirements should be:

- Specific enough for testing and checking:
- Unambiguous
- Clear
- Precise
- Self-contained

Studies has shown that 56% of the defects of a product are connected to requirements

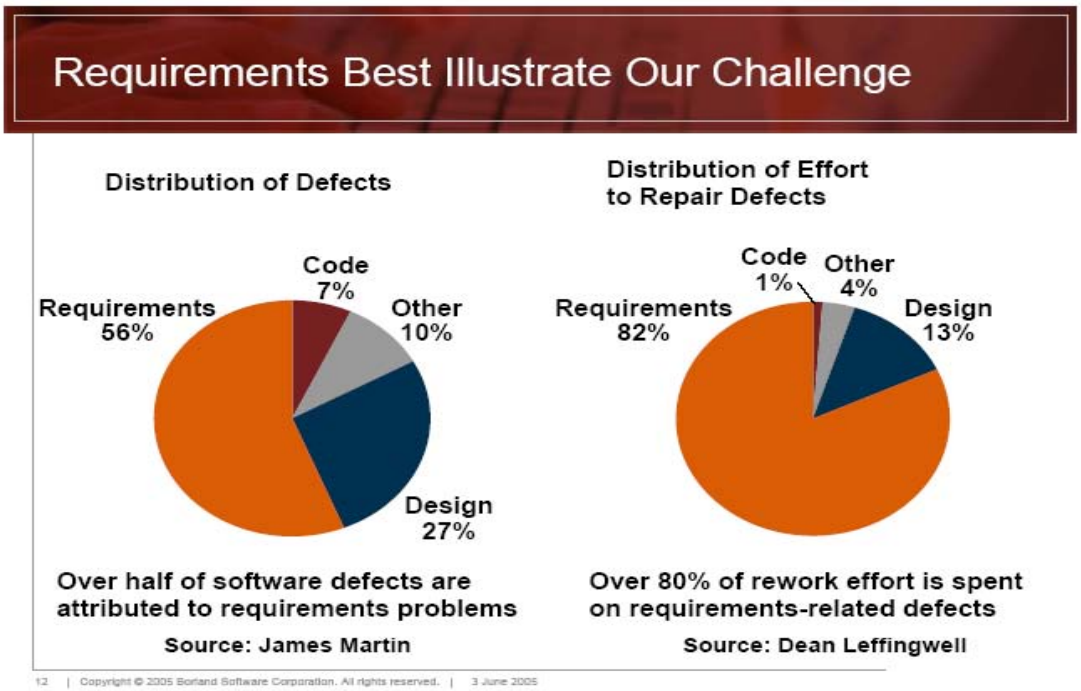


Figure 7 : Typical Distribution of defects

Requirements are often written in natural language and are source of ambiguities, inaccuracies and thus defects on the end product (e.g. use of synonyms, vague or meaningless words, forms of particular sentences).

According to the phase of the product life cycle, the correction related to a requirement defect can cost up to 200 % of the initial cost of correction if detected in upstream phase.

The analysis of requirements qualities by human means in reviews is heavy and time consuming. Numerous errors can remain undetected.
The time dedicated in reviewing linguistic aspects leave less time to analyze fundamental questions such as “are the requirements consistent and complete”, or "Is the system adequate for the operational need?"

Natural language is the most natural thing for us to do. Generally it is understood by all parties (engineer, manager, user, sponsor…).
Formal languages have been tried but have been successful in only very limited domains.
Use cases and scenarios are incomplete and difficult to partition.

- For example, an industry standard may include a significant number of sentences.
- The engineering staff may need several levels of detail.

An automated analysis of requirements enables:

- To reduce cycle time and effort with better results than those achieved by tedious manual reviews
- Early detection and correction of simple but often costly errors, leaving time to the analysts to concentrate on other value-added aspects, such as determining whether the system is really “fit-for-intended-purpose”

In a first step, we introduced in DOORS, Quality Center and MS Excel some facilities that detect forbidden words that are often cause of poor requirement formulation and are source of ambiguity because they are vague, general or not verifiable.

user friendly	fault tolerant	high fidelity
quick	rapid	fast
adaptable	flexible	easy
support	but not limited to	adequate
Minimize	Maximize	Sufficient
And	Or	etc.…

Figure 8 : Example of forbidden word

In a second step, we integrated in Quality Center a tool called LEXIOR. LEXIOR is developed by a partner of ADN: CORTIM.
LEXIOR is a tool that enables lexical analysis of requirements: it detects forbidden words but also incorrect grammatical sentences, passive voice sentences, use of wrong subjects, etc…
The analysis can be done in DOORS or through an XML file.
The tool is based on using rules for requirements writing.

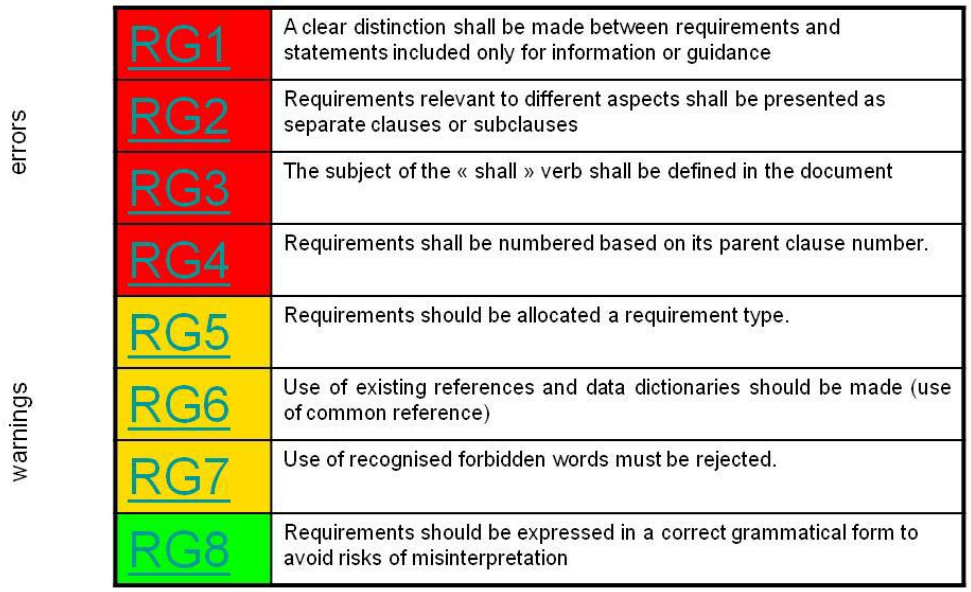


Figure 9 : Example of general rules

As the tool was already integrated to DOORS, we decided to integrate it into HP Quality Center (QC), as this tool is widely used in the pharmaceutical industry.

The integration was done to enable the analysis within the Requirements Module of the tool:

- Requirements are defined in the Requirements Module
- Within the QC GUI, the user enables the analysis of a set of selected requirements or requirements in a folder.
- The results of the analysis are raised in QC: results are stored in a folder in the Requirements module. Each rule violation and weight is traced to the related requirement.
- In the Dashboard module, metrics show the distribution of the defects and the weight
- A user can correct a requirement (or a set of requirements) and check the correctness of his changes.

Example of analysis:

- Rule RG1: Only one « requirement keyword » shall exist per requirement:
  - shall, will, should, may, can, cannot
- Requirements
  REQ1 : The instrument shall attempt frequency tracking of the signal. - OK
  REQ2 : The instrument shall provide measurements of the signal when frequency tracking has been accomplished. - OK
  Instead of
  REQ1 : The instrument shall attempt frequency tracking of the signal and, after this has been accomplished, shall provide measurements of the signal. - NOK

Traceability: Data model

Traceability is a support process of GAMP 5.
If the concept of traceability is easy to understand, it can be difficult to implement.
Traceability supports the engineering and the verification/validation activities of a project. It allows:

- To give evidence of the compliance to the regulation
- To ensure that all the needs are taken into account
- To ensure that there is no over specifications
- To optimize tests with a risk analysis support and suppliers tests plans
- To give evidence that the system has been correctly tested

In practice, traceability has to be organized. If we take for example a category 5 system from GAMP 5, we will have the following deliverables:

- User Requirements Document: this deliverable describes the requirements of the various stakeholders of the system. In practice it is organized in several documents such as Regulation requirements, Business requirements, Maintenance requirements, Security requirements...
- Functional Specification: this deliverable describes the system requirements. In theory it is one single document, but for practical reasons of organization, this deliverable is split in several documents per disciplines or functionalities.
- Design Specification: this deliverable describes the architecture and the design requirements. It can be also split in several documents.

If small projects can have a minimum of 3 requirements documents, huge project can drive to 20, 30, 60 requirements deliverables.
One can easily understand that if MS tools can be used in small projects, huge projects with many actors spread in different locations and countries need to be supported by dedicated engineering tools.
In huge projects, traceability is also more complex to establish: interactions between the different deliverables are multiple; a lot of actors are implied in the requirements definition process.
In this context it is necessary to define a data model and to plug it in the engineering tool. On a traceability point of view, the goal of this data model is to define the relationships between the data in the different deliverables of the project.
By implementing a data model in a tool, it avoids uncontrolled multiple links in several directions (“spaghetti” links) preventing impact studies, traceability analysis … It also guides the designers or the testers in defining “what to trace to what”.
For example in the data model, it’s the person in charge of the Functional Specification that responds in term of traceability to the User Requirement Document. This drives to a traceability link “satisfies” from the System requirement to the User need requirement, and not in the other way.

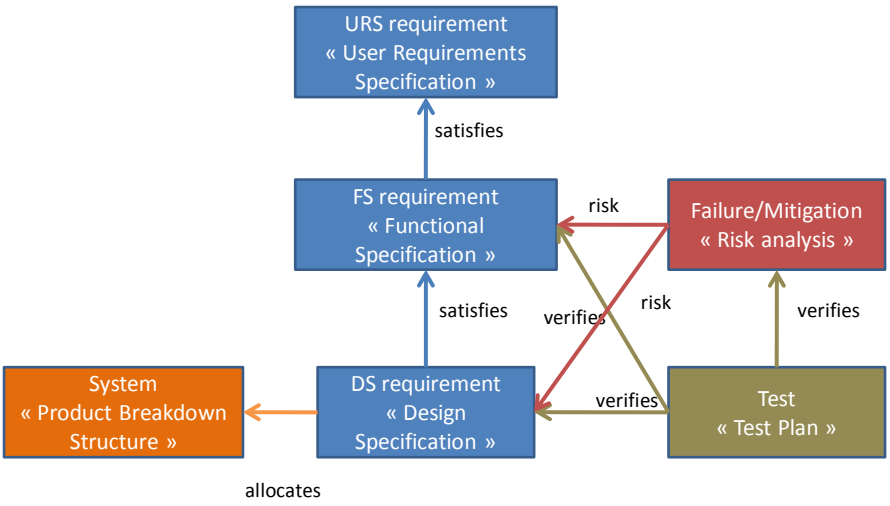


Figure 10 : Example of data model

Of course this data model has also to be defined for small projects, but it is easier to deploy because a small number of actors are implied.

Implementation in DOORS:

To integrate the GAMP 5 methodology in Doors, ADN has defined a data model according to the GAMP recommendations (see Figure 9).

Some constraints were added in DOORS to the traceability links to ensure compliance with the model data and the implementation of risk analysis as defined in GAMP. Setting the data model also allows rapid identification of:

- Requirements not identified or not traced.
- Requirements that have risks or no risk
- Requirements defined for a system in the architecture
- Tests that have requirements to verify or no requirements with a level of risk

Finally, to adapt to the pharmaceutical industry, all regulations have been incorporated into DOORS modules to be used as required in each project.

This work on Doors enabled ADN to offer our customers the pharmaceutical kit validation directly used for their validation project. This kit includes the applicable regulation elements, a pre-formatted risk analysis and list of standard tests to run.

This kit has been developed in DXL (Doors Extended Language) which is the programming language of DOORS.

Implementation in Quality Center:

Quality Center proposes a standard data model after its organization module and very close to the GAMP methodology:

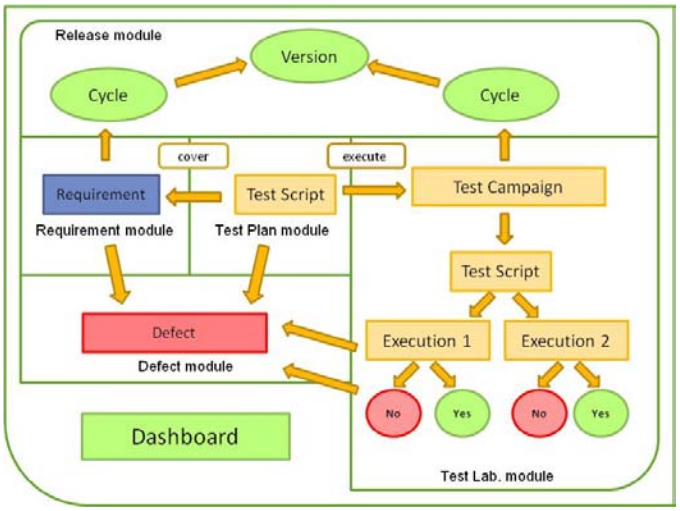


Figure 11 : Organization of data in Quality Center

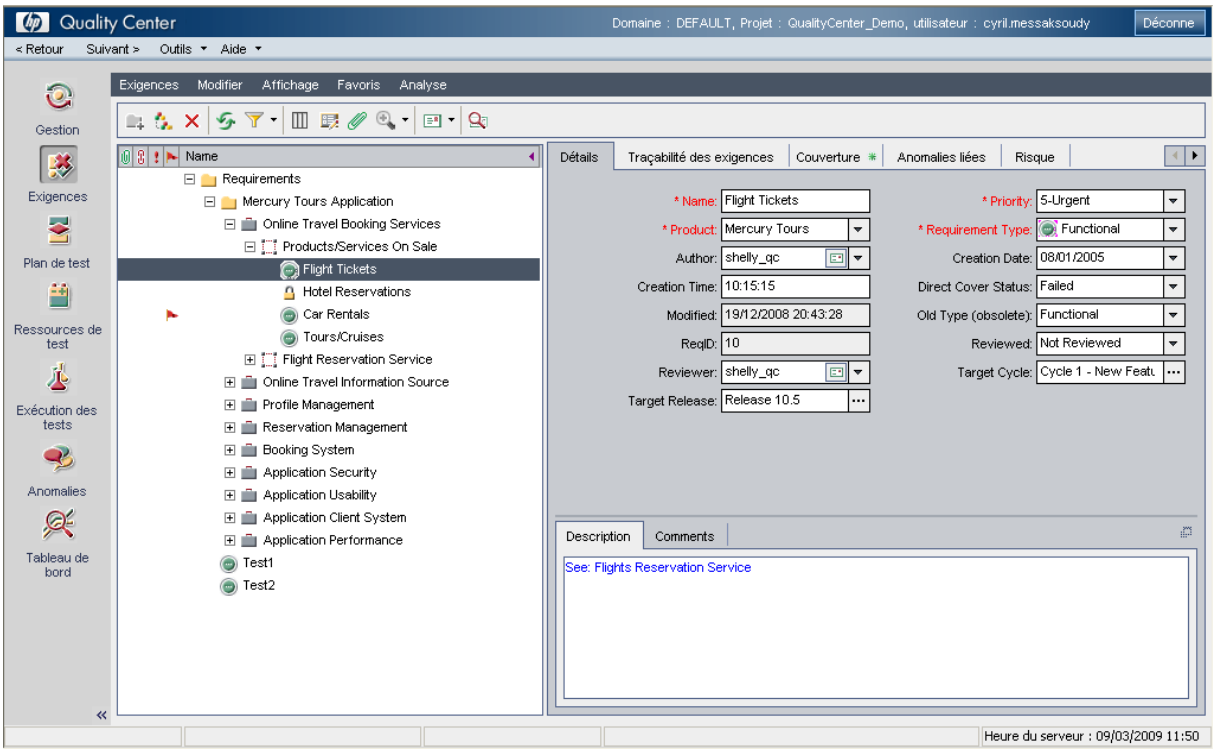


Figure 12 : Example of the Quality Center interface, Requirements module

To integrate the GAMP 5 methodology in Quality Center, we constrained the data model already present in the tool by disabling the creation of non authorized links
Like in DOORS, all regulations have been incorporated in the "requirements" module to be used as required in each project.

Beyond setting the module requirements (eg definition of properties of a requirement),
We also have developed a specific risk analysis (other than standard one delivered with the tool) authorizing the establishment of a single failure for a number of requirements. This setting authorizes to include at the requirements level the level of remaining risks defined by the mitigations.

Specific developments have also been developed to extract specifications, traceability matrices, and validation reports.

All these developments made in SQL and VBS has been a qualification and is available in SaaS (Software as a Service).

Electronic signature:

On March 20, 1997, FDA published a final rule on electronic records and signatures, a regulation that will have a profound effect on device companies. This rule (21 CFR 11) establishes the criteria under which FDA will deem electronic records and electronic signatures equivalent to paper records and traditional handwritten signatures.

Moreover, faced to a traditional process of paper record associated with handwritten signatures, most companies struggle with inefficient processes, increasing delivery delays, … non value added wastes (e.g. tests done in several countries to be signed are wastes in delays)…
The tendency is to move to a Lean thinking approach: the dematerialization of the processes – use of data base tools to manage the development process, use of electronic signatures.

In this context, ADN has developed an electronic signature in HP Quality Center. The use of such functionality enables to sign a requirement, a set of requirements, a test script, and a test campaign before and after execution and baselines. It enables to avoid paper record by keeping in the tool signed data.

The following figure shows the interface of the electronic signature facility:

The electronic signature enables to sign:

- Individual requirements (with or without requirements versioning)
- A group of requirements
- Requirements baselines
- Test campaigns before and after execution
- Individual tests scripts
- A group of tests scripts
- Individual defects
- A group of defects

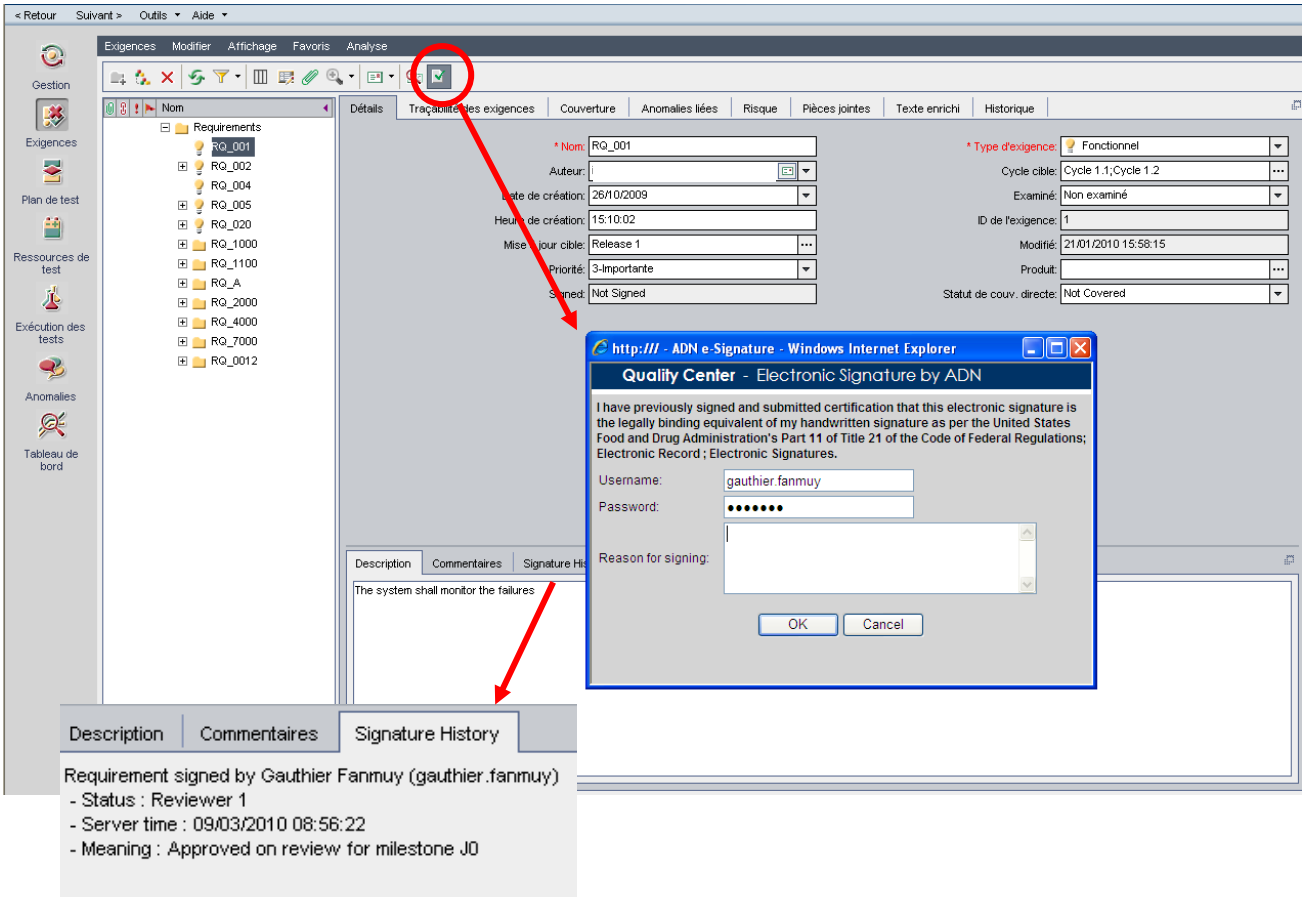


Figure 13 : Electronic signature MMI developed in Quality Center

A workflow enables multiple signatures. These signatures are logged in an audit trail available in the man machine interface.

REFERENCES

[1] GAMP 5 - Good Automated Manufacturing Practices, ISPE
[2] 21 CFR Part 11 - Electronic Records, Electronic Signatures
[3] GMP 2007 - Good Practices Manufacturing
[4] GMP -- Good Manufacturing Practices
[5] ITIL - Information Technology Infrastructure Library - V3
[6] CMMI - Capability Maturity Model Integration

BIOGRAPHY

Gauthier Fanmuy serves as Technical Director and Head of the Skill Center "Systems Engineering" at ADN (http://www.adneurope.com), a consultancy company. He has a particular responsibility for technical coordination, deployment of methods and tools related to transverse Systems Engineering and Project Management.
Within AFIS (http://www.afis.fr), he is the leader of the Technical Committee "Global Processes".
Within INCOSE he is the Leader of the “Systems Engineering for Very Small and Medium Entities WG”. He is involved in several WG such as bio-Medical, Lean and Requirements.
He previously worked at PSA Peugeot-Citroen, where he was responsible for the implementation of engineering requirements on a project vehicle. He was also responsible for the deployment of System Engineering in a department in charge of vehicles engineering and testing.
Prior to this experience, he worked at Dassault Aviation on weapons systems Rafale, Mirage 2000-9, F1CR Mirage and Atlantic 2, where he served various responsibilities in the quality, development of tactical functions, integration of opto-electronic equipment, covering all activities of the lifecycle of a system.
He is an Engineer from the Ecole Centrale de Marseille.

Richard Szczepaniak is the founding chairman of CORTIM, a consultancy company in Systems Engineering since 2001. CORTIM is specialized in requirements engineering and participates to AFIS RE working group and INCOSE RWG.
Richard has led several company-wide projects dedicated to requirements engineering process improvement, mainly using the DOORS tool. Outcomes of this kind of projects are best practices, tool customization and user training.
Before, he was Senior Expert in on-board data management for Astrium Space company after having worked on automatic control, operational support for satellite deployment and software project management.
He is a graduated engineer from Ecole Nationale Supérieure de l’Aéronautique et de l’Espace in Toulouse. He contributed to set up the Systems Engineering training program in Supaéro.