L'importanza di un processo robusto di Requirements Management
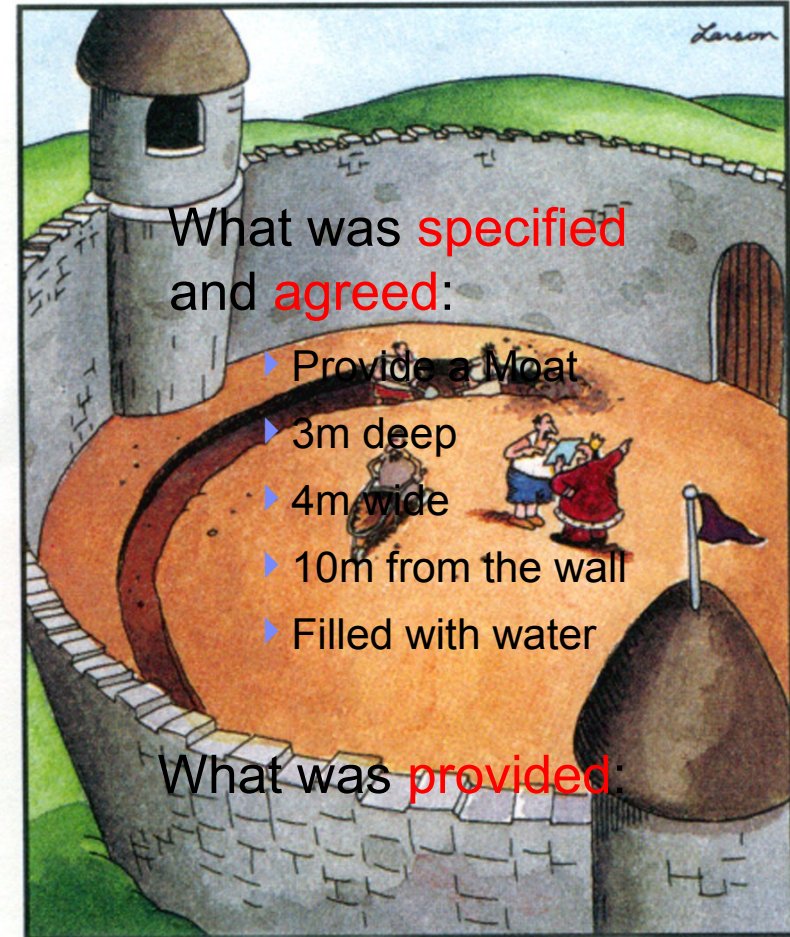
Gianluca Monticone – Senior Systems Specialist

What is Requirements Management?
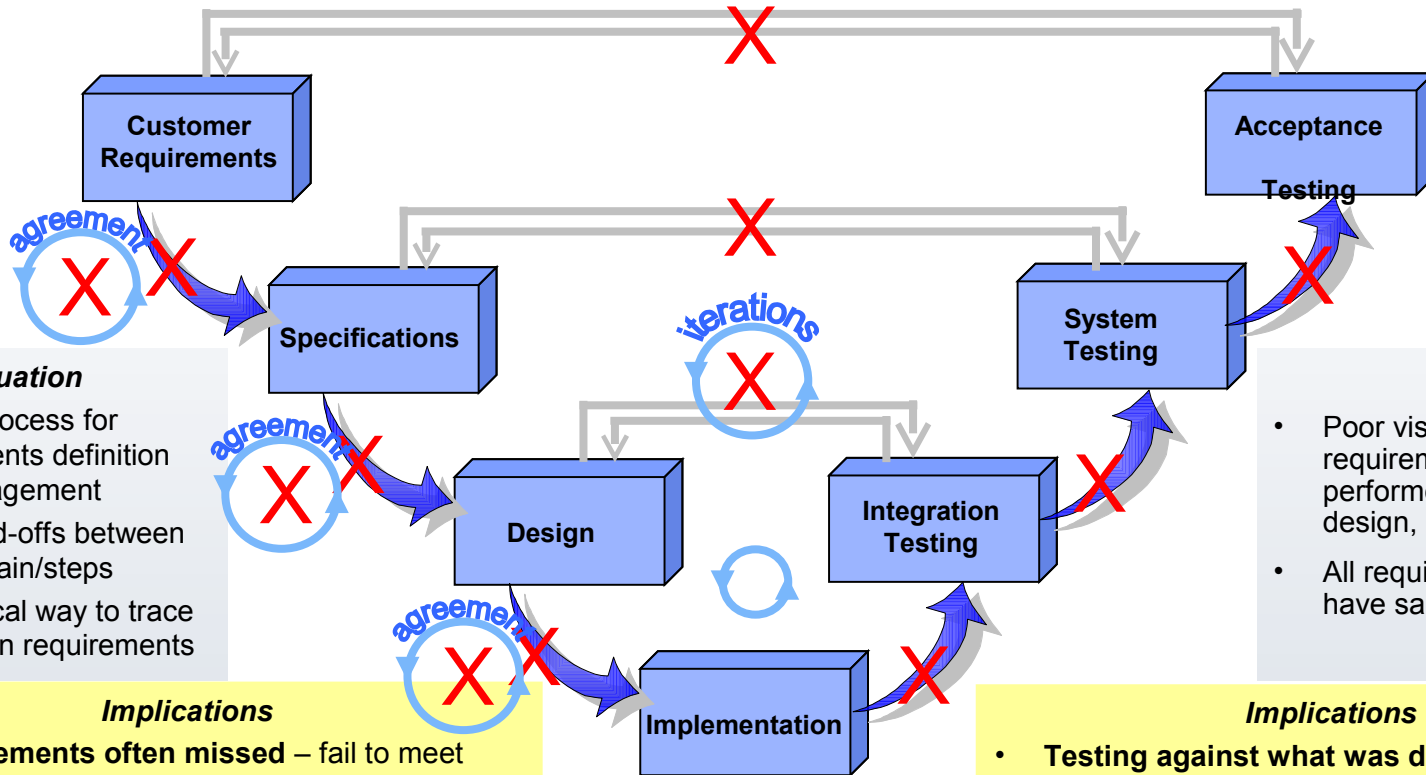
"The purpose of requirements management is to establish a common understanding between the customer and the … project … This agreement with the customer is the basis for planning and managing the … project."

The Capability Maturity Model for Software (CMM®) from the Software Engineering Institute at Carnegie Mellon University. - www.sei.cmu.edu/cmm

What was specified and agreed:
- Provide a Moat
- 3m deep
- 4m wide
- 10m from the wall
- Filled with water

What was provided:

Suddenly, a heated exchange took place between the King and the moat contractor, and hence, requirements management was born.

Today's Typical Siloed Systems Engineering Process

Customer Requirements

Specifications

Design

Implementation

Integration Testing

System Testing

Acceptance Testing

agreement

iterations

Situation

- Ad hoc process for requirements definition and management
- Poor hand-offs between supply chain/steps
- No practical way to trace changes in requirements

Situation

- Poor visibility to clear requirements so tests are performed against the design, not the requirements
- All requirements appear to have same priority

Implications

- **Requirements often missed** – fail to meet stakeholder and business needs, missed schedules and cost targets
- **Lack of traceability** – can't demonstrate compliance/V & V
- **Lack of impact analysis** – can't react to changing contractual requirements

Implications

- **Testing against what was designed, not what the customer asked for** – rework after acceptance testing fails, induces schedule slip, more cost
- **Business impact not used to drive testing** – increased business risk since high priority requirements may be pushed to end of process and dropped
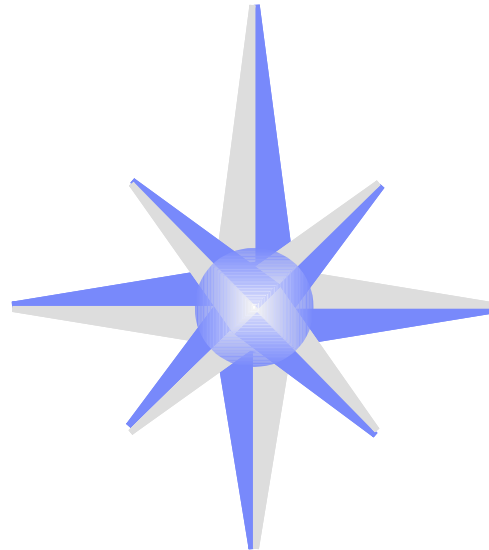
So how do I fix this mess? And stop being late and over budget?

4 Principles for Effective Requirements Lifecycle Management

Recognize the needs of all stakeholders

N

Automate your requirements process

W

E

Use abstraction to manage complexity

S

Integrate requirements across the lifecycle

Principle 1: Recognize the needs of all stakeholders

Avoid Premature Details at Top Levels

Problem	**Solution**
State what the stakeholders want to be able to do: **Capabilities**	State what the system must do: **Function**

Principle 1: Recognize the needs of all stakeholders

An Exercise in clear and concise descriptive writing?

The system shall perform at the maximum rating at all times except that in emergencies it shall be capable of providing up to 125% rating unless the emergency condition continues for more than 15 minutes in which case the rating shall be reduced to 105% but in the event that only 95% can be achieved then the system shall activate a reduced rating exception and shall maintain the rating within 10% of the stated values for a minimum of 30 minutes.

Principle 1: Recognize the needs of all stakeholders

Seven Criteria for Requirement Statements

Each requirement statement should be:

1. *Individual*: each statement is a single traceable element

2. *Unique*: each statement is uniquely identified

3. *Clear*: each statement is clearly understandable

4. *Precise*: each statement is precise and concise

5. *Abstract*: does not impose a solution on the next layer

6. *Quantified*: each statement has acceptance criteria

7. *Testable*: each statement can be validated/verified

Principle 1: Recognize the needs of all stakeholders

Document Structure

Stakeholder Requirements Specification
- 1 Introduction
  - This section should provide an overview of the entire
  - 1.1 Purpose (of the document)
  - 1.2 Scope (of the software)
  - 1.3 References
  - 1.4 Overview
- 2 Overall Description
  - This chapter describes the general factors that affect
  - 2.1 Product perspective
  - 2.2 General capabilities
  - 2.3 General constraints
  - 2.4 Operational environment
- 3 Specific Requirements
  - 3.1 Capability requirements
  - 3.2 Constraint requirements
  - 3.3 Interfaces

Structure helps:

- Understand context

- Assess completeness

- Identify repetition/conflict

- Navigate/search requirements

Principle 1: Recognize the needs of all stakeholders

Structure and Templates

Harmony/ITSW Stakeholder Requirements Specification

1 Introduction
1.1 Purpose (of the document)
1.2 Scope (of the software)
1.3 References
1.4 Overview
2 Overall Descri
2.1 Product perspe
2.2 General capab
2.3 General constr
2.4 Operational en
2.4.1 Assumptions
3 Specific Requ
3.1 Capability requ
3.2 Constraint requ

Harmony/ITSW Stakeholder Requirements Specification

1 Introduction
This section should provide an overview of the entire document and a descrip the software.

1.1 Purpose (of the document)
This section:
(1) defines the purpose
(2) specifies the intende

1.2 Scope (of the
This section:
(1) identifies the softwa
(2) explains what the pr
(3) describes relevant b
(4) should be consistent

1.3 References
This section provides a

Harmony/ITSW Stakeholder Requirements Specification

2.2 General capabilities
<Actor> shall be able to <Act> <Action_Subject> whilst <Qualification> <Qualification Subject>.

<Actor> shall be able to <Act> <Action_Subject> within <Performance> of <Event> whilst <Qualification> <Qualification Subject>.

2.3 General constrai
<Actor> must be able to <Act less than <Performance> <Un <Qualification Subject>.

<Actor> shall not be placed in regulation>.

Name	Description	Type
Admin	Harmony/SE Project Administration Folder	Folder
Analysis and Design	Harmony/SE Analysis and Design Folder	Folder
Planning	Harmony/SE Project Management Folder	Folder
Requirements	Harmony/SE Requirements Folder	Folder
Subsystems Data	Harmony/SE Subsystems Data Folder	Folder
Verification	Harmony/SE Verification Folder	Folder
Glossary	Harmony/SE Glossary	Formal

Document Structure

Boiler-plate text

Requirement templates

Project templates

Principle 1: Recognize the needs of all stakeholders

Attributes

Identification

Type

Performance

Priority

Status

Principle 1: Recognize the needs of all stakeholders

Virtually unlimited user-defined attributes

- Nearly unlimited number of attributes in a spreadsheet-like view

- Values can be calculated for metrics collection

- A value or attribute may be displayed in any column

Principle 2: Use abstraction to manage complexity

Building a Requirements Hierarchy
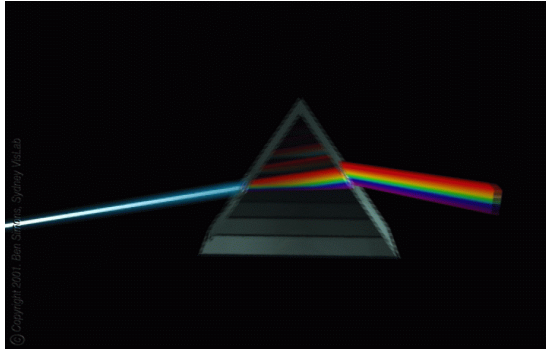
Decomposition
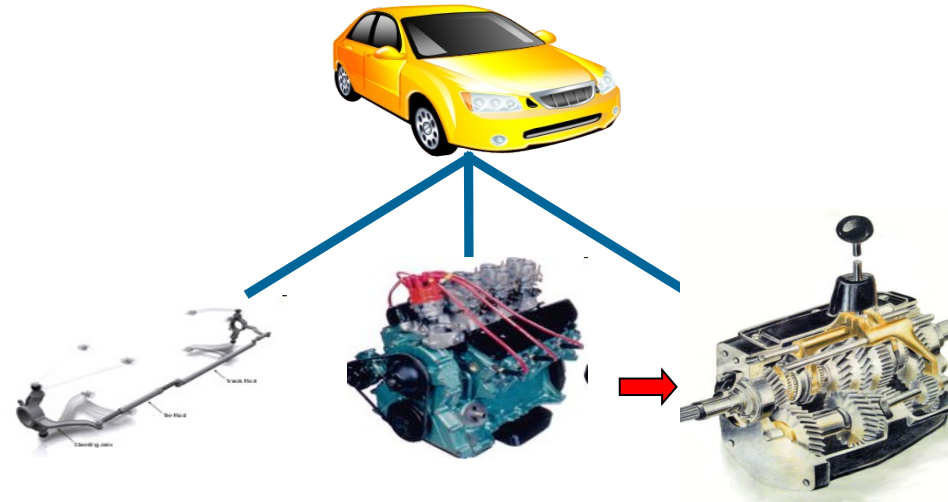
Transformation

Design-driven

Allocation

Principle 2: Use abstraction to manage complexity

Why is Traceability Important?

Why are we building this?

Where is this implemented?

How do I test this?
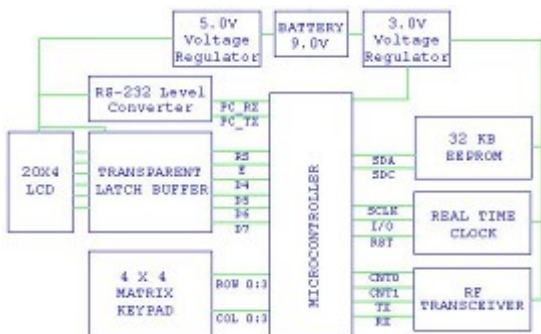
Can we show these answers? (Governance)

Principle 2: Use abstraction to manage complexity
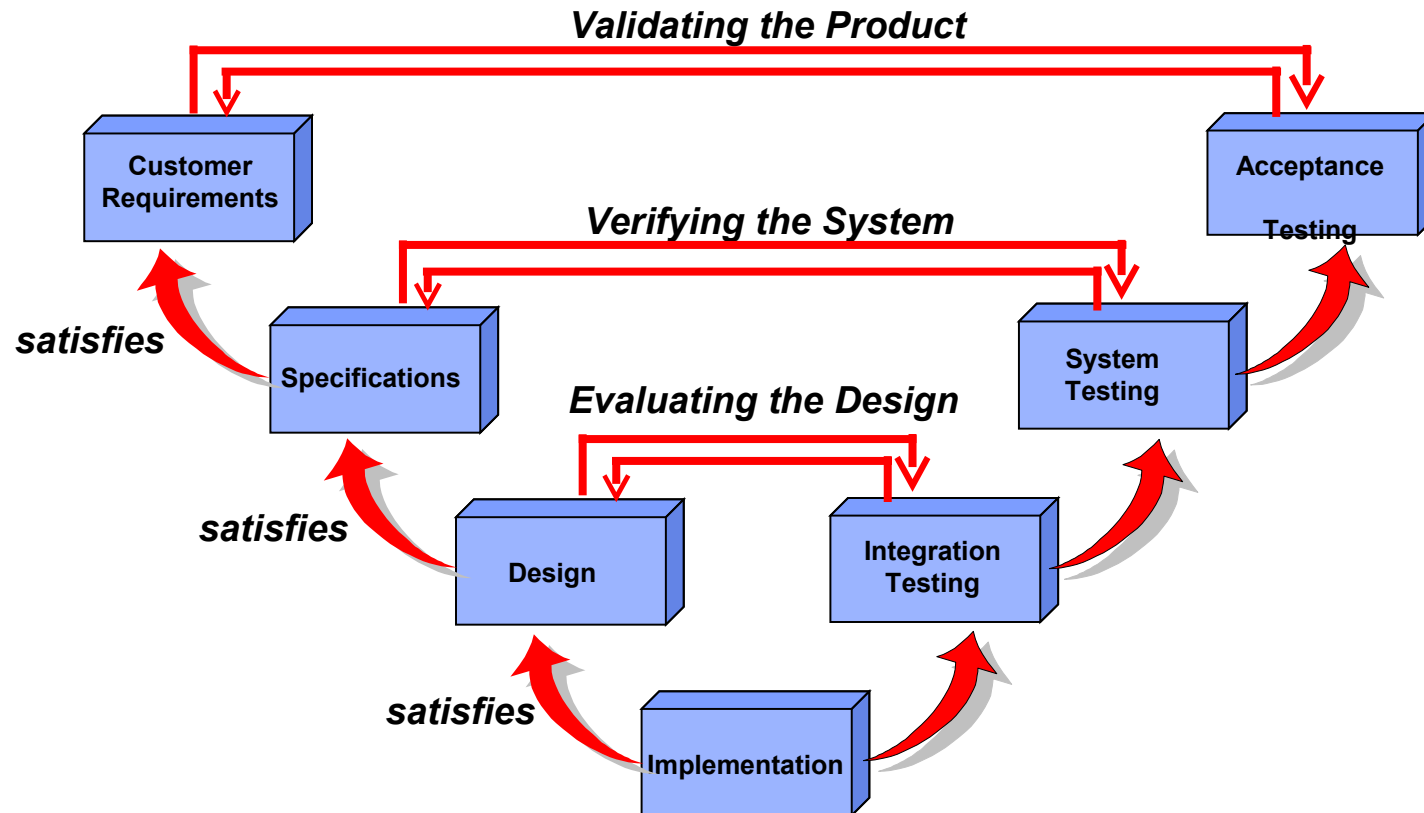
Create, review and use traceability

Principle 2: Use abstraction to manage complexity

Create, review and use traceability
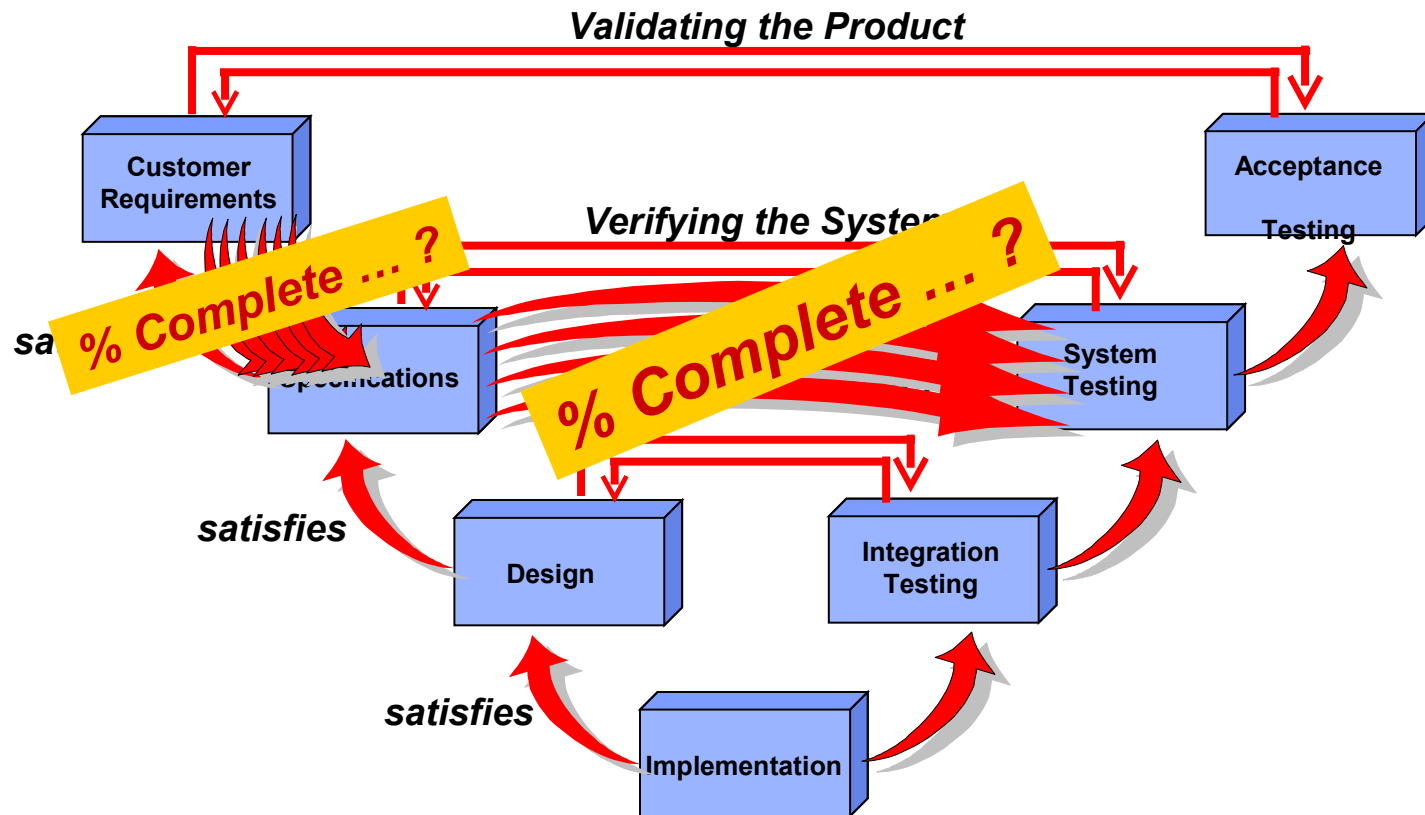
Coverage Analysis – Top Down

Principle 2: Use abstraction to manage complexity

Create, review and use traceability
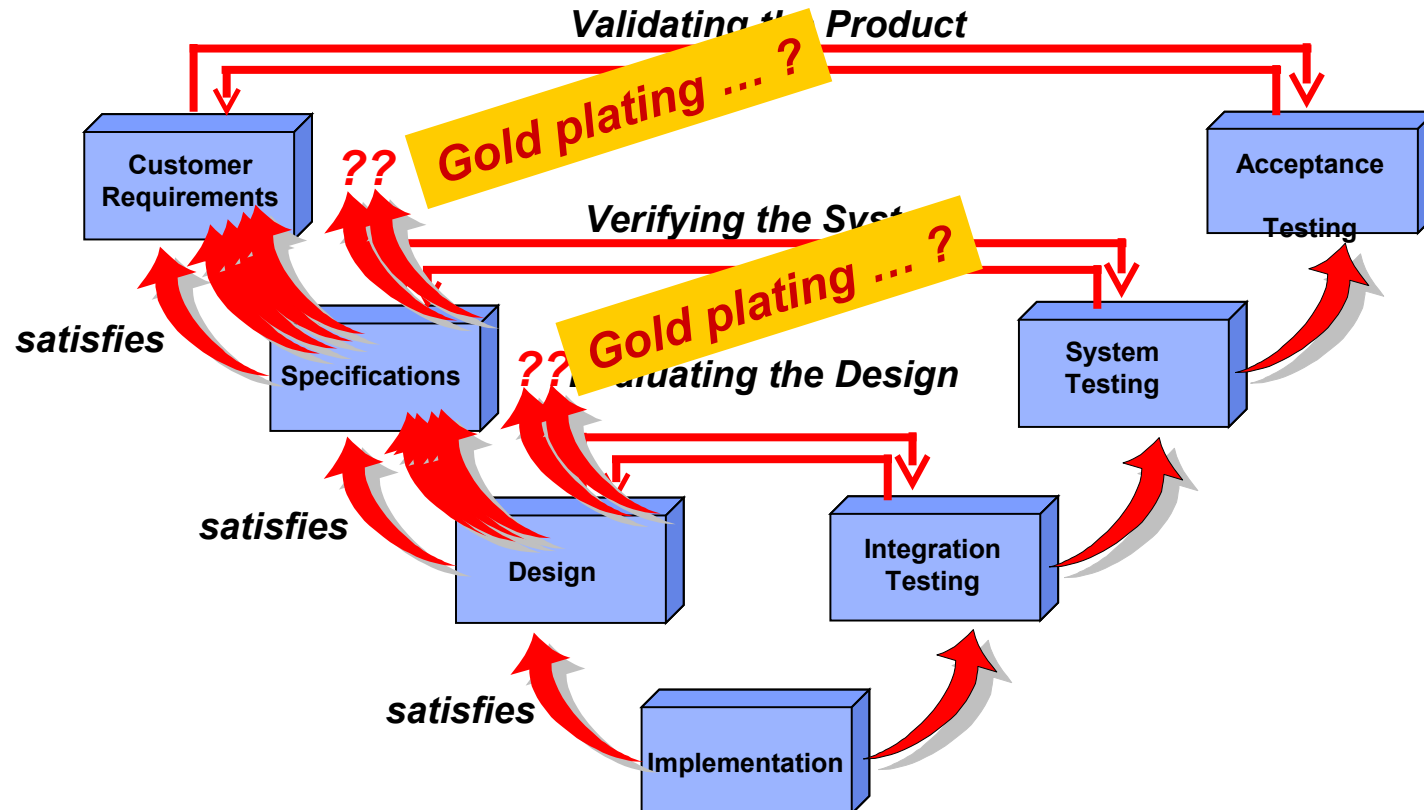
Coverage Analysis – Bottom Up

Principle 2: Use abstraction to manage complexity

Multi-Level Traceability - Info transparency allows you to take control

Complex traceability made as simple as drag and drop

Trace through multiple levels of documentation in a single display

Product Reqs　　**System Reqs**　　**Design**　　**Software Requirements**　　**Test Plans**

Make maintaining traceability an asset rather than an overhead

Principle 3: Integrate RM across the lifecycle

RM across the Enterprise

Pyramid from top to bottom:
- **Board** — Corporate Vision
- **Program Managers** — Portfolio and Program Management
- **Project Managers, Analysts and Requirements Engineers** — Project Management, Requirements Management
- **Developers** — Requirements Use, Development Tools
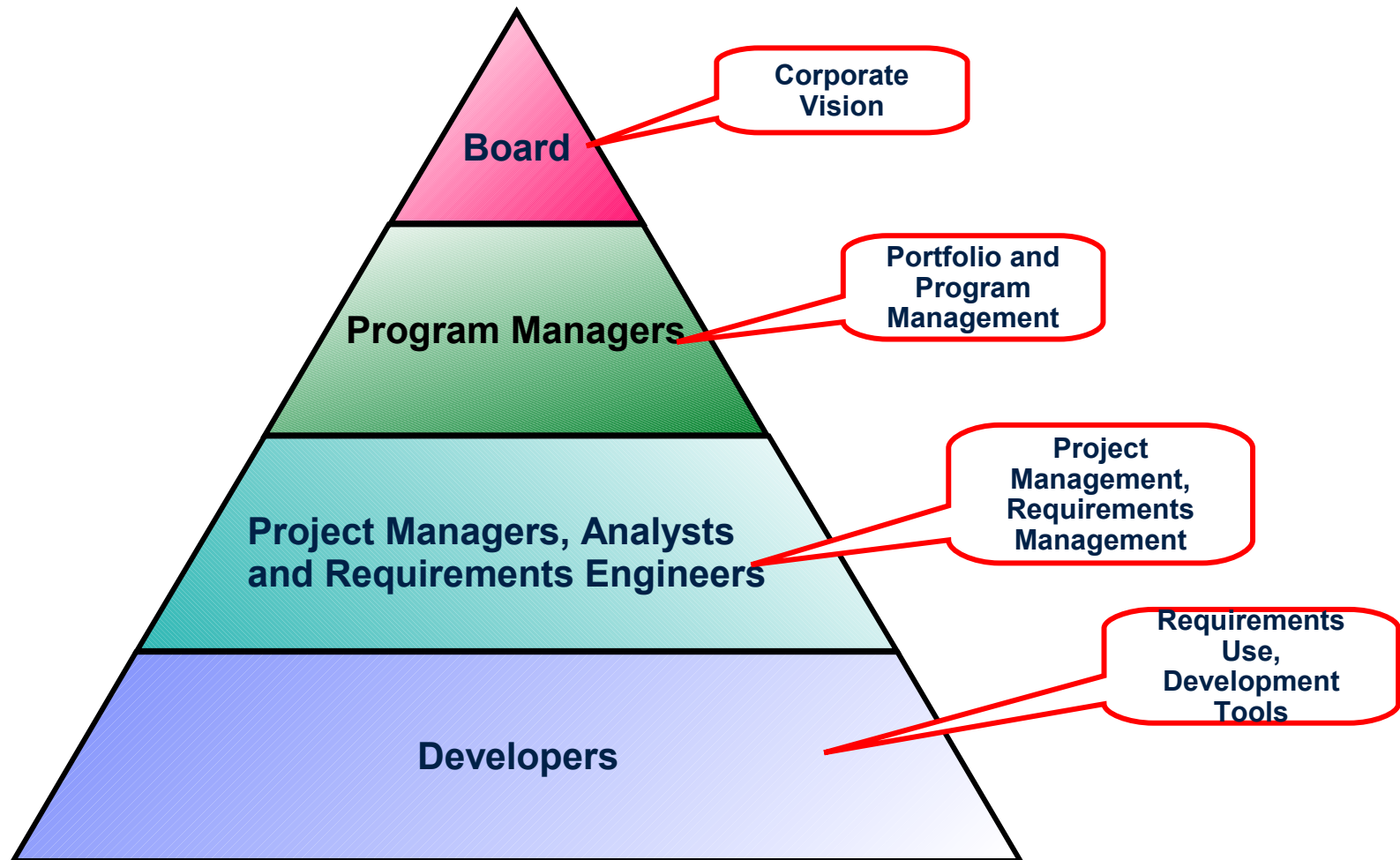
Principle 3: Integrate RM across the lifecycle
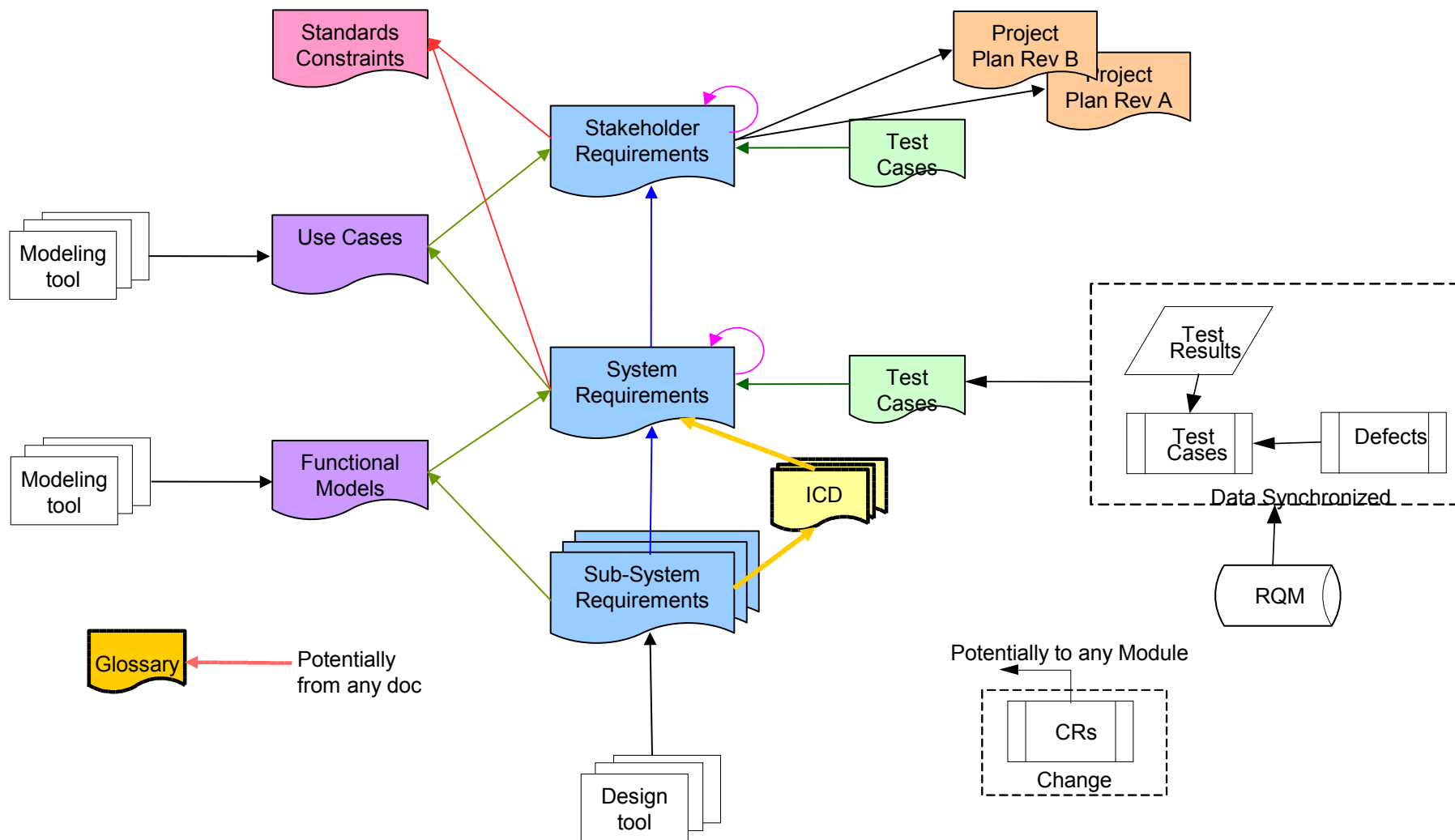
Principle 4: Automate your requirements process

Measure the requirements process

- CMMI, ITIL and other process assessment frameworks expect measurement
  - ▶ CMMI needs RM to get to level 2
  - ▶ Need **measurement** to understand efficiency and consistency
  - ▶ Key to continuous process improvement

Metrics Collection	Total NC	Total C	Agreed C	Delta NC	Delta C	Allocated NC	Allocated C	Proven NC	Proven C	Deleted NC	Deleted C
24 21/5/05 - 22/5/05											
24.1 21/5/05 - 21/5/05											
24.1.1 /01 - Period ending now - real time/Reqts - Standard view	4	13	0	0	0	3	10	1	4	0	0
24.1.2 /01 - Period ending now - real time/Reqts - 01 - Data Entry	4	13	0	0	0	3	10	1	4	0	0
24.2 22/5/05 - 22/5/05											
24.2.1 /01 - Period ending now - real time/Reqts - Standard view	0	0	0	0	0	0	0	0	0	0	0
24.2.2 /01 - Period ending now - real time/Reqts - 01 - Data Entry	0	0	0	0	0	0	0	0	0	0	0

Principle 4: Automate your requirements process

Effective Requirements Mng realizes quantifiable savings
and with a tool you are able to measure

Example: how to measure and results

- Development releases consisting of typically 8000 requirements used to take 6 months

- Phase 1 - Application of robust process and tool enforcement reduced this period to 12 Weeks over a period of 1 year

- Phase 2 - Continuous process improvement for a further 12 months reduced this period to 6 weeks

- Over time, defect removal and effectiveness was 55% at phase 1, 88% at phase 2 and still improving

- Defects undetected end up with the customer – the figures represent huge improvements in cost of re-work, quality and customer satisfaction

Principle 4: Automate your requirements process

Use a Requirements Management Tool

Document structure

Attributes

Filter to focus

View related information

View historical information

4 Principles for Effective Requirements Lifecycle Management

Recognize the needs of all stakeholders

N

W E

Automate your requirements process

Use abstraction to manage complexity

Rational DOORS

S

Integrate requirements across the lifecycle

Please note

IBM's statements regarding its plans, directions, and intent are subject to change or withdrawal without notice at IBM's sole discretion.

Information regarding potential future products is intended to outline our general product direction and it should not be relied on in making a purchasing decision.

The information mentioned regarding potential future products is not a commitment, promise, or legal obligation to deliver any material, code or functionality. Information about potential future products may not be incorporated into any contract. The development, release, and timing of any future features or functionality described for our products remains at our sole discretion.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon many factors, including considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve results similar to those stated here.

DOORS Roadmap

2011	2012	2013	2014+

DOORS 9.3.0.x DWA 1.4.0.x Fix packs

DOORS 9.4 DWA 1.5

DOORS 9.5 DWA 1.6

DOORS DWA 9.x

DOORS DWA 9.x

Interoperation

DOORS Next Gen Beta 1

DOORS Next Gen Beta 2

DOORS Next Gen Beta 3

DOORS Next Gen

DOORS Next Gen

DOORS Next Gen
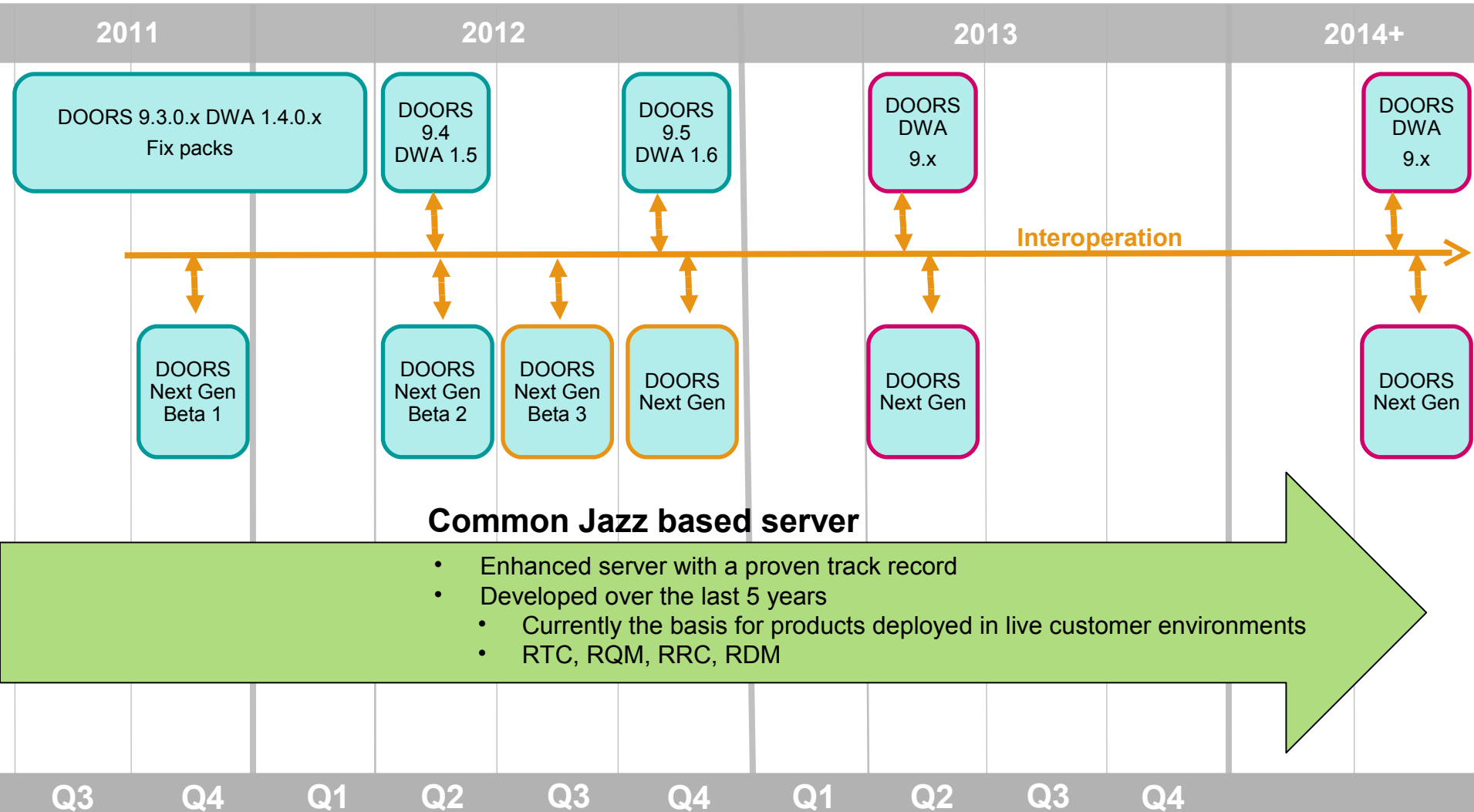
Common Jazz based server

- Enhanced server with a proven track record
- Developed over the last 5 years
  - Currently the basis for products deployed in live customer environments
  - RTC, RQM, RRC, RDM

Q3	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4

STATUS: ━ *Released* ━ *Planned* ━ *Conceptual*

Investing in current DOORS 9 deployments

2011	2012	2013	2014	2015...

IBM Rational DOORS 9.4 and DWA 1.5

- DOORS – HP Quality Center (March 2012)
- Upgrade RIF to the latest version ReqIF
  - Data exchange between DOORS 9 and DOORS Next Generation
  - Improved support for your supply chain
- Security Enhancements
  - Move authentication / authorization from the client to the DOORS server
- Transition integrations from synchronization to linked lifecycle data
  - Improved visibility of lifecycle attributes and traceability
  - New integrations to RQM, Design Manager (beta)
  - Linking between different RM databases
  - Document generation
    - Run user defined templates without an RPE license
- Usability Improvements
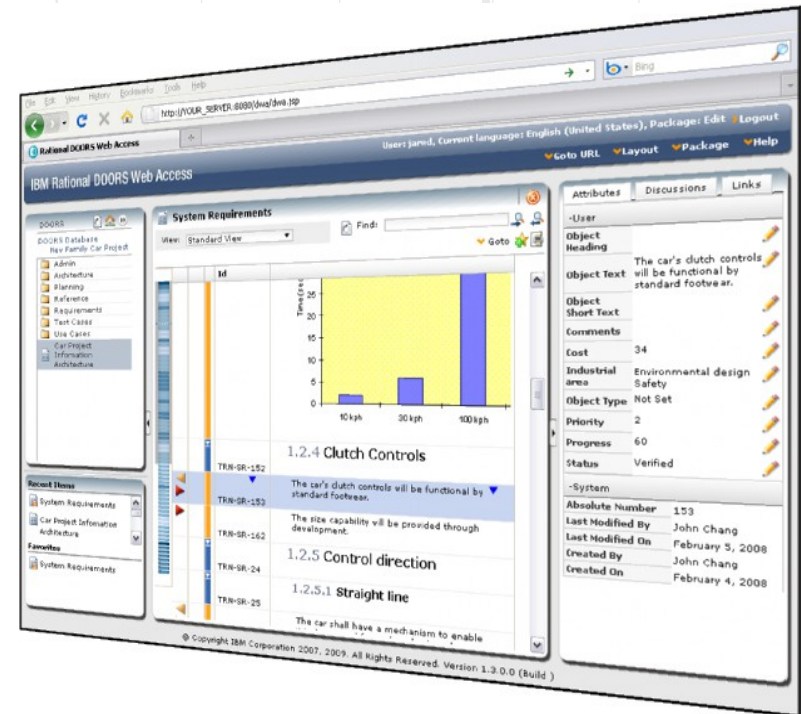
STATUS: *Released* *Planned* *Conceptual*

Future releases of DOORS 9.x –candidate themes

2011	2012	2013	2014	2015…

DOORS

- Usability Enhancements
- Reporting over system engineering metrics
- Database-wide query
- Richer OSLC Integrations
- Additional Integrations

DOORS Web Access

- Persistent user preferences
- Document generation

STATUS: ━ *Released*   ━ *Planned*   ━ *Conceptual*
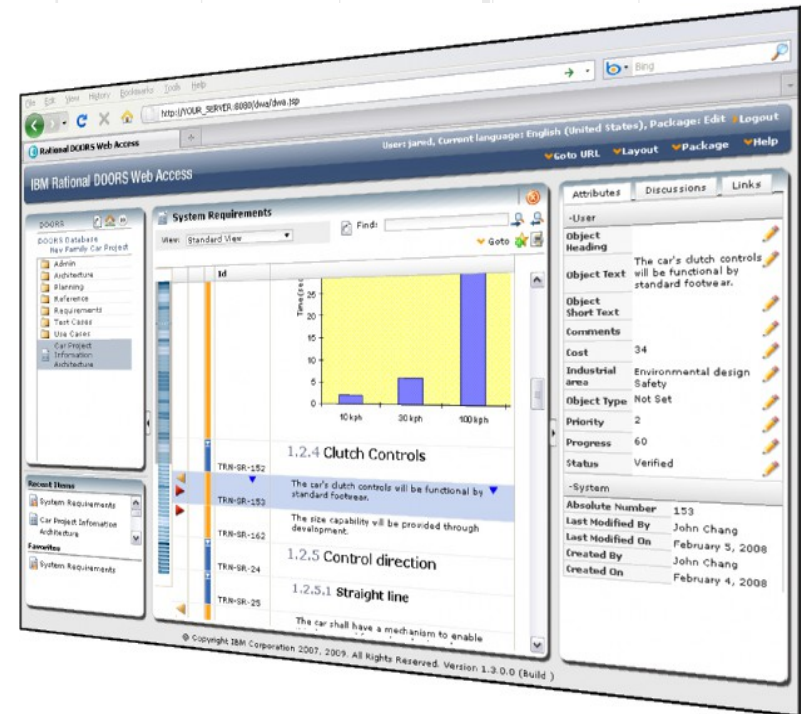
Architectural Plan for Rational RDM Tools

- Requirements visibility and traceability across the lifecycle
- Open integration architecture built on the Jazz Team Server

- Integrations using Open Services for Lifecycle Collaboration (OSLC)

DXL

Web Client   **Rich Client**

RM Services

Jazz

Team Server

COTS database

Requirements Management Services → Publish → Consume

Change Management Services → Consume

Quality Management Services → Consume

Architecture Management Services → Consume

Publishing Services → Publish

Additional Services → Publish → Consume

OSLC
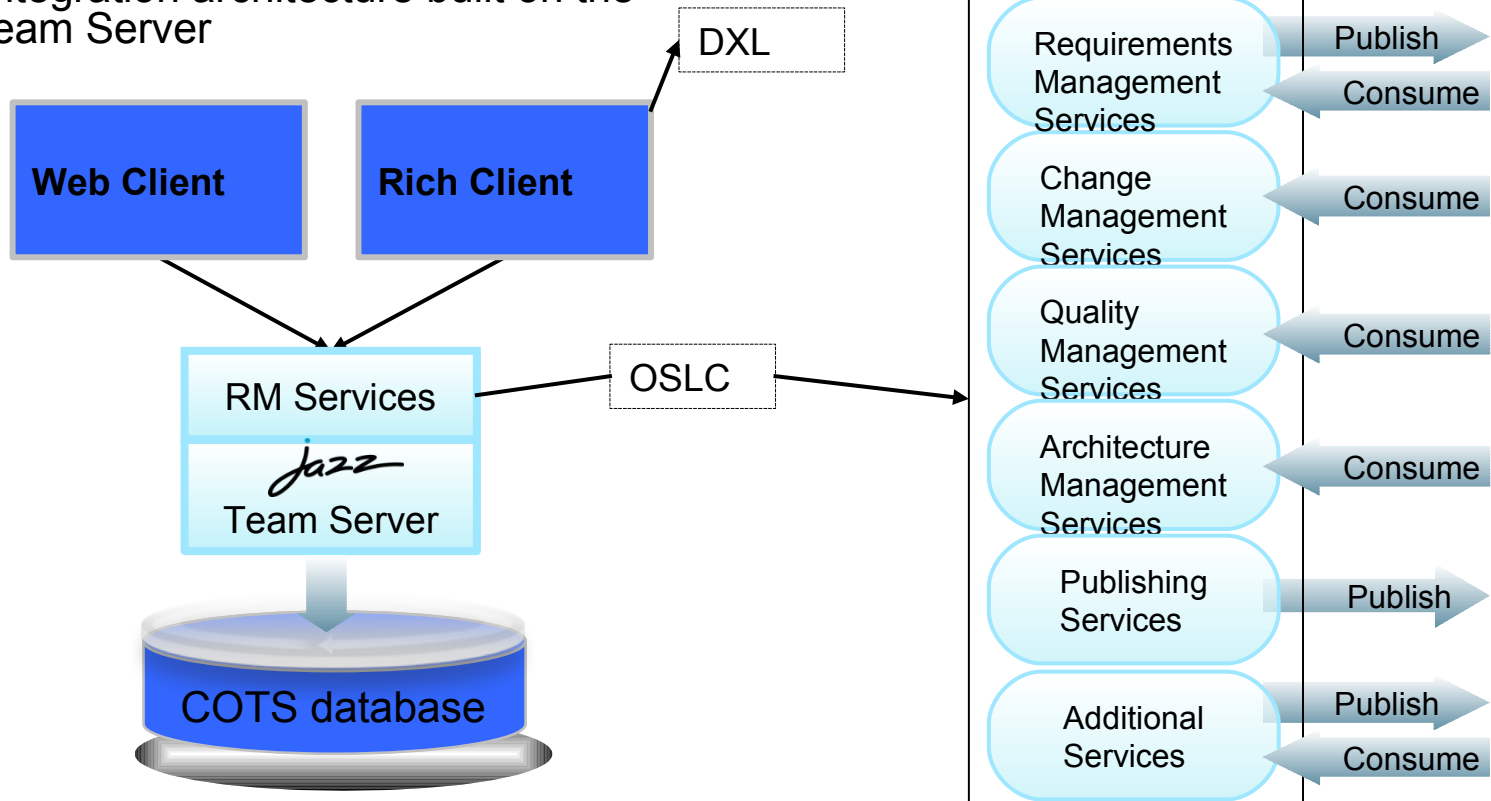
- JEE application server (Tomcat, WebSphere)
- Relational Database (DB2, Oracle, MS SQL server)
- Optional: LDAP-enabled directory server for users/groups
- Optional: clustering via WebSphere
- Browser support (various versions of IE, Firefox)
- Rich client application support varies across individual applications

Familiarity for existing DOORS users

DOORS Next Module view (rich client)

Familiar look and feel minimizes transition training for existing users

Familiarity for existing DOORS users

DOORS Next Module view (rich client)

Familiar look and feel minimizes transition training for existing users

Familiarity for existing DOORS users

DOORS Next Module view (rich client)

Familiar look and feel minimizes transition training for existing users

Fully functional Web client

DOORS Next Generation Module view (web client)

Modern web look and feel minimizes adoption cost for new and casual users

Filter Pane   **Module**   **Properties**

IBM Rational DOORS Next Generation
DOORS concepts improved and much more….

Definition
NEW!
- Rich-text documents
- Diagrams: Process, Use Case
- Storyboards, UI sketching & flow
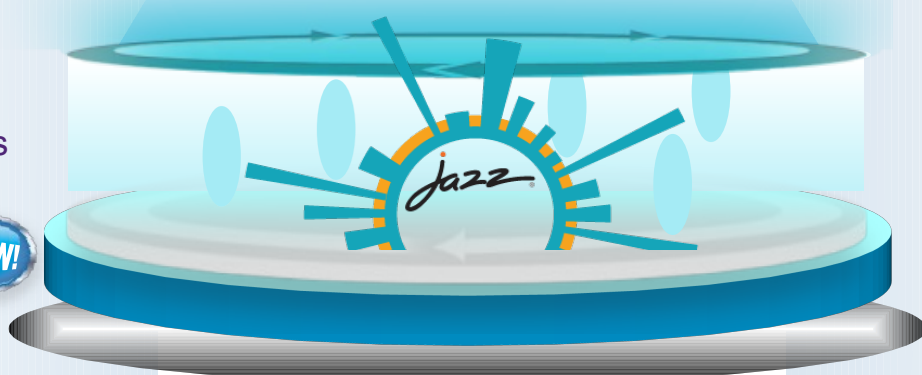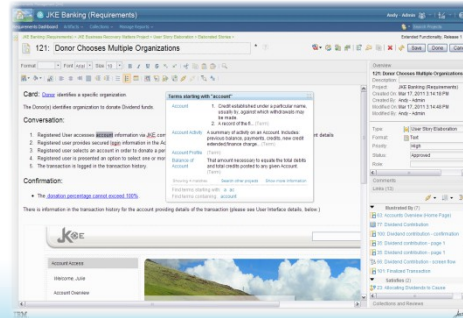- Project glossaries
- Templates

Visibility
NEW!
- Customizable dashboards
- Analysis views
- Collections
- Milestone tracking & status

Collaboration
NEW!
- Review & Approval
- Discussions
- Email Notification

Management
Improved!
- Structure, Attributes/Types
- Traceability, Filtering, Tags
- Baselines, Change History
- Reuse (reqs & types)
- Reporting Metrics & Doc.

Lifecycle
NEW!
- Central requirements, test, & development repository
- Common administration and role-based user licensing
- Warehouse reporting

Planning
NEW!
- Integrated planning
- Effort estimation
- Progress tracking

Adopt DOORS Next Generation at your own speed

Both Products can be used in parallel to ensure pragmatic low-risk stepwise migration

- **Continue to work with DOORS and DWA**

  - Gain the benefits of DOORS 9.x and future 9.x releases

jazz Building better software

Download 4.0.1 RC3 Milestone
November 13, 2012

- **Use DOORS Next Generation with DOORS 9.x**

  - "Recognizably DOORS" to aid adoption

  - Use DOORS Next Generation for new projects as it meets your needs

  - Data Import / Export between DOORS Next Generation and DOORS 9.x projects

  - Bi-directional linking between DOORS Next Generation and DOORS 9.x

  - Support for distributed RM development and Supply Chains

- **Participate in DOORS & DWA beta programs (already ended)**

  - Follow DOORS Next Generation on jazz.net

  - Participate in the beta, managed or unmanaged

  - Download milestone releases

  - Participate in development discussions

  - Submit defects/enhancements
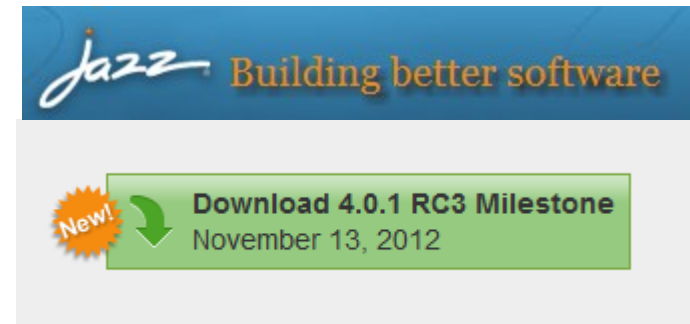
Rational DOORS Next Generation 4.0.1

Date	Type	Version
2012/11/13	Milestone	Rational DOORS Next Generation 4.0.1 RC3
2012/11/03	Milestone	Rational DOORS Next Generation 4.0.1 RC2
2012/10/19	Milestone	Rational DOORS Next Generation 4.0.1 RC1
2012/10/04	Milestone	Rational DOORS Next Generation 4.0.1 M4
2012/09/04	Milestone	Rational DOORS Next Generation 4.0.1 M3
2012/08/16	Beta	Rational DOORS Next Generation 4.0.1 Beta 4

What's next?

▪ Track 14,45 – 15,30

▶ Gestione della qualità nel ciclo di vita di un prodotto: gestione attività, requisiti, modellazione, pianificazione ed esecuzione dei test.

▪ Demo point 15,00 – 17,00

▶ Gestione tracciabilità e generazione automatica di documentazione in ambienti complessi. Generazione documentale da DOORS e Rhapsody con RPE per documenti di requisiti e architettura sistema

Further information:

"Requirements Engineering" by Hull, Jackson and Dick, Edition 2, Springer 2005

"10 Principles of Requirements Management" by Professor Ken Jackson